IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 11, 2018, accepted November 27, 2018, date of publication December 14, 2018,
date of current version January 7, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2886822

Testbed Evaluation of Real-Time Route Guidance in Inter-Vehicular Communication Urban Networks

YASER E. HAWAS[1,2], GOKULNATH THANDAVARAYAN[2], BASIL BASHEERUDEEN[2], AND MOHAMMAD SHERIF[2]

[1]Civil and Environmental Engineering, College of Engineering, United Arab Emirates University, Al Ain 15551, United Arab Emirates
[2]Roadway, Transportation and Traffic Safety Research Center, United Arab Emirates University, Al Ain 15551, United Arab Emirates

Corresponding author: Yaser E. Hawas (y.hawas@uaeu.ac.ae)

This work was supported by the UAE University Fund under Grant 31R030.

ABSTRACT This paper presents the technology and laboratory testing of an embedded system that enables inter-vehicular communication for real-time route guidance (RG). Specifically, it presents a developed prototype for testbed deployment of the vehicle-to-vehicle-based RG algorithm by Hawas and El-Sayed. For efficient communication and to prevent data overflow, the RG algorithm only allows message exchange among vehicles within the so-called geo-fence regions, which are in the vicinity of the urban network intersections. The utilization of exchanged data among vehicles for real-time navigation and best route finding follows a specific protocol and screening conditions to minimize data overflow and communication requirements. This paper reviews the detailed procedure and the mathematical formulations of the RG algorithm. Detailed functional analysis was performed on the system, and technical requirements were subsequently identified. Based on these requirements, the on-board units were designed, and the functionality was validated with different test cases. To further investigate the performance of the RG algorithm in the real world, different laboratory scenarios were formed to emulate various field conditions. The description of each scenario and the results are discussed in detail.

INDEX TERMS Decentralized systems, geo-fence, intelligent transportation systems (ITS), operational and field testing, real-time route guidance, vehicle-to-vehicle communication.

I. INTRODUCTION

Traffic congestion is one of the most common urban transportation problems and can cause massive losses of time and fuel. To alleviate congestion, implementing traditional methods such as extending infrastructure and building new roadways are sometimes undesirable and infeasible due to the limitations of space and budgets. Smart real-time route guidance (RG) is becoming one of the most popular research topics in solving congestion problems. Unfortunately, in terms of deployment of these smart RG systems, there is a disparity between state of the art in research and state of the art in practice. Intelligent transportation system (ITS)-based RG algorithms employ advanced computation and communication technologies to predict the dynamic congestion levels in a network and to depict the shortest travel path for every vehicle to reach its destination. Many challenges are faced in the deployment of such smart RG solutions. In this paper, we highlight these challenges and illustrate how such systems can be developed and tested.

RG algorithms are commonly classified in literature as either centralized or distributed [1]–[3], with many variations based on algorithmic procedures, detection and communication requirements, and control facilities, to name a few. In a centralized architecture, a single traffic management center (TMC) predicts traffic congestion evolution and provides consistent path guidance to vehicles [4], [5]. Although the centralized approach upholds a thorough and accurate analysis of traffic information, some limitations have been reported. Due to the dynamic nature of the transportation network, the TMC easily gets overloaded with information transferred from many participants, which demands vast computation, storage capacity, and communication capability. Since TMC resources are limited, the system cannot handle massive data processing, which results in performance deterioration [6]. As thousands of vehicles are connected with the TMC to receive specific RG updates (i.e., unicasting), the centralized architecture encounters challenges with network dynamics, which can eventually stress the backhaul links and

create congestion in the wireless network [7]. If the TMC is deployed to cover relatively large areas (e.g., an entire city), it is likely that the information delivery will suffer from latency and delay issues, which may affect the efficiency of the time-critical RG. Furthermore, these systems have been reported to have high operational costs [8] and be more prone to TMC failures [9]; if the TMC fails, the entire system will stop functioning.

In contrast, distributed architectures have been proposed as an alternative to operate real-time reactive strategies that rely on limited information. Because information is processed in a distributed way, the system requires less computational power and data storage (see [3] for more details). Since the RG algorithm is localized and performed at the edge of the network, the response times on the control actions are significantly improved. Also, distributed systems create less overhead or overflow since the communications are limited within the local controllers' territories [10]. These distributed architectures have been further classified as non-cooperative and cooperative systems. In a non-cooperative distributed system, the local controllers are distributed to provide reactive RG for vehicles based on the limited information extracted from the local controller's territory only. In non-cooperative systems, the local controllers work independently without being connected with one another. Hawas and Mahmassani [1] proposed a non-cooperative decentralized structure and a family of simple heuristic-based rules for a reactive real-time RG system. The system was envisioned as a set of local controllers distributed in the network to sense vehicle information such speed and density and to utilize this information to guide vehicles within the local controller's territory. As hardware units, the local controllers (placed at intersections) extract some traffic measures within their specified space. Such traffic measures were utilized in determining routing decisions. As the controllers were non-cooperative, the system had the potential drawback of vehicle cycling.

Distributed cooperative systems play a primary role in resolving the cycling problem encountered in non-cooperative systems by enabling communication and cooperation between adjacent decentralized local controllers. Hawas [3] presented a decentralized cooperative system for real-time RG. The system performed knowledge exchange between adjacent local controllers to evaluate the alternative sub-paths emanating from the decision node towards the destination. Another distributed approach for vehicle routing, which uses a multi-agent system, was reported in [11]. In the cooperative scheme, the system adopts a vehicle-to-infrastructure (V2I) communication for RG. For routing decisions, the infrastructure agents (local controllers) were utilized to provide predictive traffic density information for vehicles to compute the shortest route to the destination. These cooperative systems provide reliable RG to vehicles and avoids the cycling seen in non-cooperative systems. However, cooperative systems require additional technological infrastructure to enable data transfer among the local controllers (infrastructure to infrastructure communication), in addition to the essential communication between local controllers and their territory vehicles (two-way communication V2I). With these communication requirements, cooperative systems naturally have limitations as they impose higher implementation costs and have scalability issues. For example, when a cooperative system extends its RG coverage to other locations, additional infrastructure hardware is required to shield the new areas, leading to an increase in deployment cost. Moreover, this requires establishing connectivity between the new and existing local controls, which means reconfiguring the local controllers and leads to increased operation and management costs.

To the best of the authors' knowledge, no well-developed normative centralized or decentralized RG methodologies (with explicit real-time RG to vehicles) have been tested or implemented in the field. These systems require data centers or local controllers and communication infrastructure (with high cost). Few existing systems, such as Google Maps [12] and TomTom [13], can be regarded as real-time centralized RG systems, but the accuracy of such systems has not been verified or reported by researchers in the field. It is also not clear how these systems collect vehicular data and estimate the link travel times that are essential to estimating shortest routes. Initial field assessments by the authors of this study indicated some considerable variations between travel times estimated by Google Maps and actual travel times, especially during peak hours.

To enable RG deployment in real time, one must design a system that is least dependent on TMC hardware support or V2I communication. This is only possible through inter-vehicular communication (IVC) or VANET technologies. Using VANET technology, vehicles perform a distributed coordination with other vehicles using vehicle-to-vehicle (V2V) communication for many safety [14] and end-user application systems [15]. The literature includes different domains of V2V technologies (e.g., cooperation [16], routing [17], clustering [18], platooning [19], and data dissemination [20]) due to its flexible nature of adapting to different network topologies without any additional infrastructure support. Compared with other centralized and decentralized systems, the V2V-based RG algorithm could be easily deployed at a lower cost.

A recent extension to the distributed cooperative system was the introduction of the so-called inter-vehicular communication-based RG algorithms, which enabled information exchange between vehicular entities at specific locations only when some screening criteria were satisfied [10], [21]. The communication between any two vehicles was initiated and exchanged data were used for real-time navigation if and only if some specific conditions were met (as will be reviewed in the following section). V2V algorithms may have differences, such as the extent of communication range, frequency of the exchanged data, and algorithmic processing methodology, based on the adapted communication standards [22]. Nonetheless, incorporating V2V-based RG algorithms entails the least communication requirements

(compared to centralized or decentralized distributed V2I systems). The classification of different architecture metrics for RG algorithms is outlined in Table 1.

TABLE 1. Classification of the existing architectures for RG.

Architecture	Controllers	Advantage	Disadvantage
Centralized	Single Traffic Management Center (TMC)	Global perception, Improved theoretical accuracy	Latency, Delay, Congestion, Single-point-of-failure
Distributed Infrastructure Based	Non-Cooperative Local Controllers	No additional Infrastructure-2-Infrastructure (I2I) messages	Vehicle cycling
	Cooperative Local Controllers	No vehicle cycling	Scalability issues, Extra I2I messages
Distributed VANET Based	Cooperative Vehicles	Cost effective, Scalable, Minimum message exchange	-NIL-

Previously, two different IVC-based RG algorithms (named forward and backward) were developed and tested in a simulation environment and compared against the centralized and decentralized RG algorithms reported in [21]. The simulation results in [21] were promising and encouraging to implement a third IVC-based algorithm with lesser communication requirements and more efficient communication protocol [10]. The third IVC-based RG algorithm in [10] was tested through simulation, and it was proven to be easily implemented in a real-time environment and reduce communication complexity, processing time, and bandwidth.

This paper reviews the mathematical formulation of the IVC-based RG algorithm developed by Hawas and El-Sayed [10] and the details of the hardware technology used to implement it. We also discuss the performed lab test results. Section II summaries related research. Section III reviews the IVC-based RG algorithm by Hawas and El-Sayed [10]. Section IV introduces the system requirements for implementing the algorithm in real-time. Section V discusses the system design, and Section VI presents the functional testing. Section VII discusses the test-bed scenarios and system evaluation, and Section VIII briefly discusses one performed real-time field test. Finally, Section IX concludes this paper.

II. RELATED LITERATURE WORK

In the centralized RG architecture, most of the existing work (e.g., [4], [5], [7]) was impractical for real-time implementation and may require high operational costs due to the complexity and additional hardware support required. There are other online centralized services such as Google Maps [12] and TomTom [13] that provide normative routing suggestions on a macroscopic level. In these systems, all vehicles with same origin and destination would be instructed to take the same route at a specific time instance, which might shift the congestion but may not distribute it. There are no published articles on how Google Maps collects and processes individual vehicular data or links traffic conditions, but apparently these infrastructure-based systems analyze the data from phone GPS devices to estimate the travel time from source to destination. For this purpose, the service adapts massive centralized data centers, which demand massive computational power and storage. Moreover, these online RG models have not been verified by researchers, nor have their accuracy levels been compared or studied with the other existing systems.

In the literature, inter-vehicular communications have been predominantly used to perform several network management functions and for various ITS applications (e.g., [22]–[24]). In particular, the V2V-based RG is one of the ITS applications based on important factors, including nature of information and knowledge being shared among vehicles. The nature of information with dynamic real-time route update by Ding *et al.* [25] provided a reliable RG and served better than the static mechanism that worked principally on the shortest path algorithms (e.g., Dijkstra) using fixed historic parameters (Wei and Meng [26]). The knowledge shared among vehicles should precisely extract routing information with minimum possible information size. When routing information is shared at the global knowledge level, the information size becomes higher and consumes more processing power. With an increase in participants sharing global information in the network, congestion will occur in the network due to its higher data rate. On the other hand, when the routing information is shared at the local knowledge level, simpler and more feasible RG can be achieved within a short coverage range.

To develop an RG model with reduced computational and communication overhead, distributed V2Vs are amongst the most promising technologies to process information at the edge of the network without a centralized infrastructure or hardware support [27]. For example, Ding *et al.* [25] presented a V2V-based real-time RG algorithm that adapts a limited-scope flooding strategy to find potential shortest routes and bypass void areas. The shortest travel time was estimated by disseminating the route query (RQ) and route reply (RR) packets in the network. However, this scheme does not update the routes of vehicles during trips. Likewise, Pan *et al.* [28] developed a V2V-based distributed re-routing strategy to avoid congestion. The system adopted a fixed threshold value to select the least congested lane, which may not be feasible for urban roadway environments. Another threshold-based V2V re-routing scheme was proposed in [29]. This system used a trust probability to decide whether the current guided route remains the shortest travel time. For this purpose, a vast data collection was performed, which required much storage and a complex data structure. Overall, the existing systems in the literature require more data processing and massive wireless communication bandwidth allocation. Moreover, none of these

systems have actually been deployed for real-life testing and verification.

Unlike other existing systems, Hawas and El-Sayed [10] devised an autonomous real-time RG algorithm, which included the best factors of dynamic nature of information and local knowledge. When compared with the existing inter-vehicular systems, this algorithm does not create any overhead in the network or data overflow, and it operates at minimal cost. In the following section, we review the mathematical formulation of the algorithm and its functionalities through a network example.

III. ALGORITHM

The proposed algorithm allows information exchange among vehicles at some locations previously specified in the network (e.g., real life physical intersections or mid-link virtual nodes). Namely, the exchanged information by each vehicle includes its identification number, its origin node, the reversed trajectory of its path, the travel time on its current link, and the travel time along the reversed trajectory of its actual path to its current location. This exchanged information is utilized to estimate the shortest travel time path of any vehicle from its current location to its destination.

The methodology of the RG algorithm using V2V communication can be represented by a network graph model $A = \{\boldsymbol{N}, \boldsymbol{L}\}$, where $|\boldsymbol{N}|$ and $|\boldsymbol{L}|$ are nodes and links that represent the intersections and lanes in the network, respectively. The set of nodes and links are denoted as $\boldsymbol{N} = \{1, 2, \ldots i, .. |\boldsymbol{N}|\}$ and $\boldsymbol{L} = \{1, 2, .., l, .. |\boldsymbol{L}|\}$. Each node i is defined by its Cartesian coordinates: $i = (x_i, y_i) \in \boldsymbol{R} \times \boldsymbol{R}$. Each link $l = (i, j)$ is defined by its upstream and downstream nodes: $i = (x_i, y_i)$ and $j = (x_j, y_j)$, respectively. The vehicles in the network are defined by $\boldsymbol{V}_t = \left\{\boldsymbol{v}_t^l, \forall l \in \boldsymbol{L}\right\}$, where $\boldsymbol{v}_t^l$ is the number of vehicles on link l at time t. The Cartesian coordinates of any vehicle v at any time t are given by $x(v, t)$ and $y(v, t)$, such that $x(v, t)$, $y(v, t) \in \boldsymbol{R} \times \boldsymbol{R}$.

When vehicle v enters the network and starts its trip at any time t, it has an initial travel time $t_o(v)$ and an assigned path $\boldsymbol{P_o}(v)$ with an expected minimum travel time $T_o(\boldsymbol{P_o}(v))$ from a specific origin node $O(v) = (x(v, t_o), y(v, t_o))$ to the destination node $D(v)$. This initial path $\boldsymbol{P_o}(v)$ and the corresponding travel time $T_o(\boldsymbol{P_o}(v))$ may be estimated using a suitable shortest path algorithm (e.g., Dijkstra) based on some historical information in the network.

At any node i and time t, $\boldsymbol{v}_t^{(k,i)}$ and $\boldsymbol{v}_t^{(j,i)}$ denote the set of vehicles along links (k, i) and (j, i) at time t, respectively. The links (k, i) and (j, i) belong to the set of incoming links to node i: $\boldsymbol{l}_i^-$. Similarly, the links (i, k) and (i, j) belong to the set of outgoing links of node i: $\boldsymbol{l}_i^+$. We define $\bar{v}_t^{(k,i)}$ to be a subset of $\boldsymbol{v}_t^{(k,i)}$ ($\bar{v}_t^{(k,i)} \subseteq \boldsymbol{v}_t^{(k,i)}$) that includes only the vehicles that have already reached and queued within the communication range (geo-fence) of node i at time t.

The geo-fence at node i can be represented by the area trapped between two circles (centers are located exactly at node i) whose radii are G_i^n (inner smaller radius) and G_i^x (outer larger radius), respectively, where $G_i^n < G_i^x$. Vehicles are allowed to communicate if and only if they are located between these two circles close to node i. If $G_i^n = 0$, vehicles can communicate within the larger circle whose radius is equal to G_i^x, and they can continue communicating until they reach node i itself. The definition of the geo-fence by the two circles is crucial to the implementation of the algorithm. Setting the inner radius to a value higher than zero will enable vehicles the necessary time and chance to utilize the communicated information to actually alter the route and make necessary maneuvers to align themselves to the lanes for the newly estimated path movement. The general definition of candidate vehicles $\bar{\boldsymbol{v}}_t^{(k,i)}$ is the set of vehicles within the geo-fence of node i. That is, $v \in \bar{\boldsymbol{v}}_t^{(k,i)} | v \in \boldsymbol{v}_t^{(k,i)}, G_i^n \leq [(x(v, t) - x_i)^2 + (y(v, t) - y_i)^2]^{0.5} \leq G_i^x$. For simplicity, for the remainder of this paper, we shall assume that the internal geo-fence radius is set to zero (i.e., $G_i^n = 0$) and the external geo-fence radius $G_i^x = G_i$. In this case, where the internal geo-fence is set to zero, the set of candidate vehicles within the geo-fence will be represented as follows:

$$v \in \bar{v}_t^{(k,i)} | v \in \boldsymbol{v}_t^{(k,i)}, 0 \leq [(x(v, t) - x_i)^2 + (y(v, t) - y_i)^2]^{0.5} \leq G_i \quad (1)$$

A searcher vehicle within a geo-fence receives specific information from candidate vehicles if a specific screening condition is met (namely, only if the candidate vehicles are within the geo-fence of the closest intersection to the searcher vehicle). The screening condition minimizes the communication bandwidth and reduces data overflow.

The geo-fence radius (vehicles' configured communication range) should be dynamically set to account for the current traffic characteristics at the corresponding intersection to enable more accurate data exchange. For example, the geo-fence radius should preferably be increased when free-flow traffic conditions prevail and it should be decreased at higher congestion levels. In other words, a trade-off should be sought. When the radius G_i is set to a small value, say $G_i < 10$ *meters*, the system allows any candidate vehicle v to share its complete current link travel time information (which is most accurate), but the searcher vehicle v would have a limited chance to maneuver to an alternate path within the close proximity of node i. More congestion implies more changes to vehicular travel times over shorter distances. This is why it is more accurate to use a small radius to minimize the error (the difference between the exchanged travel time information and the actual one). On the other hand, by setting the geo-fence range to a maximum limit, say $G_i = 200$ *meters*, the searcher vehicle v can easily maneuver to alternate paths. This larger geo-fence radius is more suited to low congestion levels.

At any node i and time t, for any vehicle v in the queue of link (k, i), $v \in \bar{\boldsymbol{v}}_t^{(k,i)}$, the vehicle possesses information on its actual path trajectory $\boldsymbol{P}^-(v)$ from its origin $O(v)$ to node i. The vehicle has also some registered information on the remaining part of the path $\boldsymbol{P}^+(v)$ from i to the destination

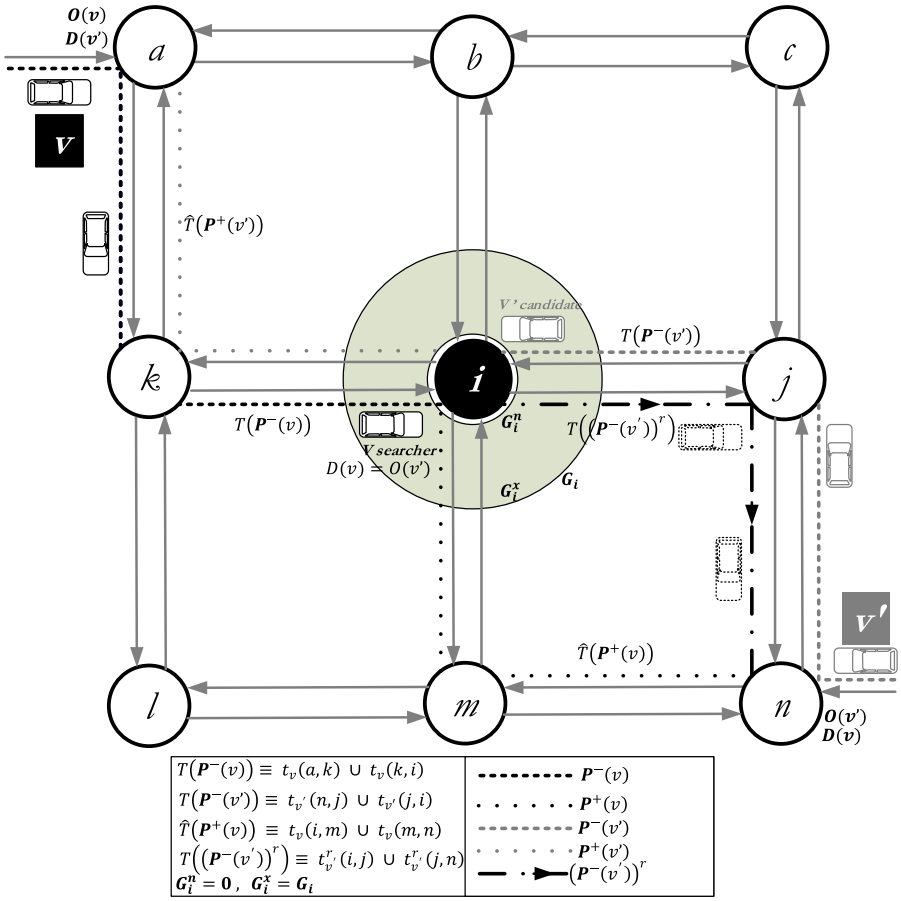


FIGURE 1. The mechanism of information exchange between a searcher and candidate vehicle.

node $D(v)$. If vehicle v does not alter its *original* assigned path until it reaches i, then $\boldsymbol{P_o}(v) \equiv \boldsymbol{P}^-(v) \cup \boldsymbol{P}^+(v)$.

In addition to registering the actual and remaining subsets of the path, the vehicle v registers the sum of actual travel time on all the links along the actual path trajectory $T\left(\boldsymbol{P}^-(v)\right)$ and the expected travel time on the remaining part to its destination $\hat{T}\left(\boldsymbol{P}^+(v)\right)$. Vehicle v also registers the actual travel time it spends along its current link $t_v(k,i)$.

With reference to Fig. 1, the proposed IVC algorithm allows for information exchange among any *searcher* vehicle $v \in \bar{\boldsymbol{v}}_t^{(k,i)}$ (which is currently on link (k,i) queued at node i at time t) and all the other queued vehicles at node i on all links other than (k,i): $\bar{\boldsymbol{v}}_t^{(j,i)} \forall (j,i) \in \boldsymbol{l}^-, j \neq k$. We identify a *candidate* vehicle $v\prime \in \bar{\boldsymbol{v}}_t^{(j,i)}$ as any vehicle queued at node i on any link (j,i) at time t whose origin node $O(v)$ is equal to the destination node $D(v)$ of the *searcher* vehicle $v \in \bar{\boldsymbol{v}}_t^{(k,i)}$. The set of all *candidate* vehicles of a *searcher* vehicle v is denoted by $\boldsymbol{V}'(v)$, where

$$\boldsymbol{V}'(v) = \left\{ v\prime \,\middle|\, v\prime \in \bar{v}_t^{(j,i)}, (j,i) \in \boldsymbol{l}_i^-, j \neq k, O\left(v'\right) = D(v) \right\}, \quad \forall v \in \bar{v}_t^{(k,i)} \tag{2}$$

Each *searcher* vehicle $v \in \bar{\boldsymbol{v}}_t^{(k,i)}$ receives some data from each *candidate* vehicle $v' \in \boldsymbol{V}'(v)$ that is registered by v throughout its trip from its origin node $O(v\prime) = D(v)$ to node i. It is important to note that each vehicle, based on its current location, origin, and destination, can be regarded as a *searcher* as well as a *candidate*. That is, as shown in Fig. 1, vehicle v is regarded as a *candidate* vehicle of the *searcher* vehicle v if both are queued at i at time t and $D(v) = O(v)$. On the other hand, vehicle v is regarded as a *candidate* vehicle of the *searcher* vehicle $v\prime$ if both are queued at i at time t and $D(v\prime) = O(v)$.

Each vehicle $v\prime$ registers (with continuous updates) its actual travel time spent on its current link when it reaches the downstream node (before it exits). With reference to Fig. 1, the actual travel time that the candidate vehicle v spent on its current link (j,i) is denoted by $t_{v\prime}(j,i)$. It should be noted that the candidate vehicle v itself, while crossing node j (*as a searcher vehicle*), would register the travel time along link (i,j) from (*candidate*) vehicles along (i,j) and queued at j. We denote this travel time by $t_{v\prime}^r(i,j)$. It is important to emphasize that while vehicle v does not actually travel link (i,j), it can still capture the link's most recent travel time while crossing node j. As a rule, any vehicle $v\prime$ along any link (j,i) can register its own $t_{v\prime}(j,i)$ when it reaches the downstream node i. Furthermore, it captures the travel time of the opposite (or reversed r) link direction along (i,j),

$t^r_{v\prime}(i,j)$, from other candidate vehicles at j when $v\prime$ crosses the upstream node j.

The *searcher* vehicle v will register the data items received from the *candidate* vehicle $v\prime$ and utilize them in enhancing the remaining part of the trip to its destination node $D(v)$. The specific data items to receive from each of the potential candidate vehicles along the incoming links to i at time t are:

1) The ID of candidate vehicle v and its origin node $O(v)$.
2) The reverse of the trajectory path $(\boldsymbol{P}^-(v'))^r = \{(i,j)\ldots\ldots.., O(v')\}$.
3) The travel time on link (i,j) registered by the candidate vehicle v at node j, $t^r_{v\prime}(i,j)$.
4) The travel time along the reversed trajectory of the actual path of the candidate vehicle v, $T\left(\left(\boldsymbol{P}^-\left(v'\right)\right)^r\right) = \sum_{(i,j)} t^r_{v\prime}(i,j), \forall (i,j) \in \left(\boldsymbol{P}^-\left(v'\right)\right)^r$. This item is captured by vehicle v by summing the travel times of the links opposite to its direction of travel.
5) The actual current link travel time of the candidate vehicle v on link (j,i), $t_{v\prime}(j,i)$.

Each vehicle v (as a *searcher* vehicle) travels from its origin $O(v)$ to its destination $D(v)$ while attempting to improve its initially prescribed shortest path $\boldsymbol{P_o}(v)$. It is assumed that the prescribed path $\boldsymbol{P_o}(v)$ is known and previously estimated using a suitable shortest path algorithm. Here, we used the well-known Dijkstra's algorithm.

Among all vehicles $\boldsymbol{V}'$, the *candidate* vehicle v^* with an actual subpath $\boldsymbol{P}^-(v^*)$ (from its origin $O(v^*)$ to node i) is identified. The vehicle v^* with the reversed sub path of $\left(\boldsymbol{P}^-\left(v'\right)\right)^r$ has the least travel time $T\left(\left(\boldsymbol{P}^-(v^*)\right)^r\right)$.

$$T\left(\left(\boldsymbol{P}^-\left(v^*\right)\right)^r\right) \min_{v\prime\in\boldsymbol{v\prime}} \leftarrow T\left(\left(\boldsymbol{P}^-\left(v'\right)\right)^r\right) \tag{3}$$

The searcher vehicle v follows its prescribed path $P^+(v) \equiv P_o(v) - P^-(v)$ from node i to the destination $D(v)$ until it meets a *candidate* vehicle v^* whose registry of $T\left(\left(P^-(v^*)\right)^r\right)$ is lesser that the previously estimated travel time on $\hat{T}\left(P^+(v)\right)$. In such a case, vehicle v is assigned to the reversed sub path of vehicle $v^*\left(P^-(v^*)\right)^r$ as the new sub path to its destination. This newly assigned sub path becomes the new sub path for vehicle v:$P^+(v)$. For more information on the algorithm and its pseudo code, please refer to Hawas and El-Sayed [10].

To demonstrate the process, we consider the network shown in Fig. 1. The searcher vehicle v originated the trip at node a and it has destination node n. Vehicle v starts its trip at time t with an assigned path $\boldsymbol{P_o}(v) = [a,k,i,m,n]$ and expected travel time $T_o(\boldsymbol{P_o}(v))$, i.e., $T_o([a,k,i,m,n])$. At node i, vehicle v meets v, which has an origin node n. Based on the exchanged information from the candidate vehicle v, the vehicle v reevaluates its remaining path to the destination node n and selects one of the following:

1) The remaining part of its own previously assigned path $\boldsymbol{P}^+(v)$ with an estimated travel time of $\hat{T}\left(\boldsymbol{P}^+(v)\right) = \hat{T}(i,m,n)$.
2) The reversed trajectory of the actual path of the candidate vehicle $v\prime$, $\left(\boldsymbol{P}^-\left(v'\right)\right)^r$, with travel time $T\left(\left(\boldsymbol{P}^-\left(v'\right)\right)^r\right) = T(i,j,n)$.

Among these two paths, vehicle v selects the one with lesser travel time to the destination node n. That is, if $T(i,j,n)$ is less than $\hat{T}(i,m,n)$, (i,j,n) becomes the new remaining path for vehicle v and $\boldsymbol{P}^+(v)$ is updated ($\boldsymbol{P}^+(v) = (i,j,n)$). If $T(i,j,n)$ is more than $\hat{T}(i,m,n)$, (i,m,n) remains the path for vehicle v. The process of information exchange is repeated at each node where communication is permitted and with all candidate vehicles $\boldsymbol{v\prime} = \left(v\prime \in \boldsymbol{V}' \mid O\left(v'\right) = D(v)\right)$.

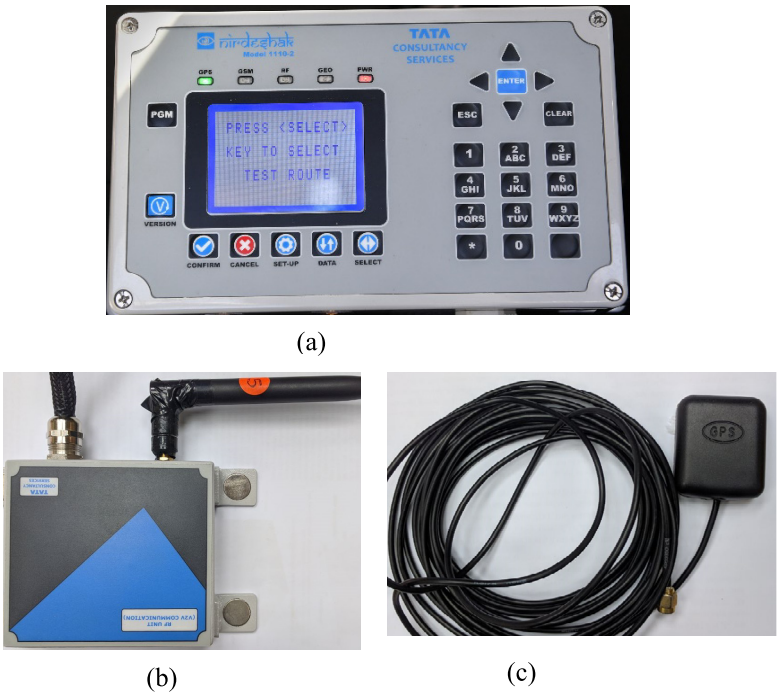
IV. SYSTEM FUNCTIONAL REQUIREMENTS

To deploy and test the effectiveness of the RG algorithm, the conventional vehicle in the real-world should be equipped with an on-board unit (OBU) to perform the inter-vehicular communications and other RG algorithm operations. The specific OBU configuration requirements, regarding the software and hardware components, were designed based on the input, processing, and output algorithm functionalities.

Based on the OBU input functionality requirements, inputs should feed to the OBU from two different data; sensed and derived information. Regarding the sensed input, the current vehicle information, such as location, time, and speed should be sensed directly from the vehicle itself. For this purpose, a global positioning system (GPS) module should be incorporated in the OBU to sense the vehicle's location. The GPS module should be compatible with other modules in the OBU and should be capable of sensing vehicle information accurately in real-time. Whereas in the derived input, information is obtained from the incoming links' vehicles traveling in opposite directions at particular locations (i.e., the geo-fence) by enabling inter-vehicular communications.

To perform these V2V operations, a wireless transmission module should be incorporated in the OBU to enable message exchange between vehicles at particular geo-fence locations. Moreover, the wireless transmission module should operate on a reliable spectrum and should hold a reliable frequency bandwidth to avoid problems such as connectivity, interference, and mobility issues.

For the processing functionality requirement, the information obtained from the input unit should be processed according to the algorithm procedure, and the desired RG output should be derived. The processing module should have a higher computing clock rate to handle the real-time information. It also requires an integrated flash storage memory to store the computed results. Moreover, with regard to the output functionality requirement, the computed RG information should be displayed as an output to the user. For this purpose, an LCD unit with a good pixel rate should be incorporated in the OBU to display the result. Besides, a user interface with different key sets is essential to allow the users to interact with the system and perform necessary operations defined in the



(a)

(b) (c)

FIGURE 2. Main modules of the on-board unit.

RG algorithm (e.g., selecting initial "predefined" route trip information).

Finally, a stable firmware must be developed for the OBUs to manage all the assembled auxiliary units and perform the RG algorithm operations more efficiently without any compatibility issues.

V. SYSTEM CONFIGURATION

In this section, we discuss the system design of the OBU hardware and firmware configurations, which were meticulously developed from the functional requirements described in Section IV. As shown in Fig. 2, the OBU has three auxiliary modules: the main processing module, wireless communication module, and GPS module. These modules were developed separately and integrated into a single OBU unit using serial ports. A firmware was developed to manage the individual hardware modules and to create a user-friendly interface. The OBU unit assembly, hardware, and firmware specifications are further detailed in this section.

A. ONBOARD UNIT (OBU) ASSEMBLY

The central processing module shown in Fig. 2(a) performs all the RG algorithm operations, and it is comprised of an LCD, a keyset, and LEDs. The LCD is used for output visualization, where each keyset enables some specific operation(s) to be carried out and to navigate in the OBU. The LEDs are used to indicate the specific operation currently being performed by the OBU and the successful connection with other auxiliary modules. The GPS module shown in Fig. 2(b) senses the vehicle's location and sends it to the central processing module. The wireless communication module shown in Fig. 2(c) enables inter-vehicular communication and the exchange of information (derived from other candidate OBUs in the same geo-fence).

The developed OBU can be installed in any conventional vehicle, as shown in Fig. 3. For receiving information, the GPS and wireless communication module are placed on the roof of the vehicle, and the central processing unit is fixed inside the vehicle. Finally, the power source (cable) of the OBU is connected to the vehicle battery. The OBUs were specifically designed to accept a varied input voltage range.

B. HARDWARE INTERFACE

To provide reliable computation power for processing the real-time information, the OBU's main processing module was designed on the NXP LPC 2478 microcontroller with a 32-bit ARM processor. The chipset has a 512 kB on-chip high-speed flash memory that enables the OBU to execute instructions at the maximum system clock rate of 72 MHz. To display the output, the central processing module was integrated with an LCD graphical display of 8 lines $\times$ 21 characters (128 $\times$ 64 pixels) to enable user interface functionalities. To capture the current vehicle information accurately, the GPS module was equipped with a U-Blox GPS sensor, which features 50 GPS/GLONASS/SBAS channels with 3.5 m CEP. For a reliable V2V information exchange, a wireless communication module with ISM 2.4 – 2.4835 GHz frequency, enabled with the uplink and downlink data rate of 250 kbps, was integrated. Finally,

FIGURE 3. OBU installed in a conventional vehicle.

TABLE 2. Obu hardware and communication specifications.

Hardware	Description
Processor	NXP LPC 2478 microcontroller, 32-bit ARM processor
ROM Memory	512 kB
LCD Display	8 lines x 21 characters at (128 x 64 pixels)
GPS	with U-Blox, 50 GPS/ GLONASS/ SBAS channels with 3.5 m CEP
Communications Radio Unit	ISM 2.4 – 2.4835 GHz frequency band
Data rate	20 Kbps

for efficient information storage, the OBU was integrated with 4 GB flash storage memory, and a database was created accordingly. The OBU specifications are summarized in Table 2.

C. FIRMWARE INTERFACE

To enable an uninterrupted stable condition of the different hardware modules (without encountering compatibility issues), a firmware was developed using a C program. Initially, the program was coded and compiled by a C compiler to obtain the object file. Then, the object file was converted into a hexadecimal form using an integrated development environment (IDE) software called Keil [30]. Finally, the generated hexadecimal file was fed to the OBU using the Flash Magic software [31]. The developed firmware enables the GPS and wireless communication modules to connect with the central processing module. In the firmware, the GPS was configured with a baud rate of 9600, 8 data bits, no parity, one stop bit to read the GPS strings and extract position, speed, date, and time. The firmware also configures the wireless communication module with a baud rate of 115200, 8 data bits, no parity, and one stop bit for a reliable V2V message exchange.

VI. FUNCTIONAL TESTING: ON-BOARD UNIT (OBU)

Upon developing the OBUs with the specifications defined in Section V, their functionalities and working conditions were tested before installing them in conventional vehicles. The OBU functional tests varied from basic operations to an advanced level of testing all the complex RG algorithmic operations. Passing the test successfully would make the OBU eligible for instalation in a vehicle. The various OBU tests were grouped into four different categories as detailed further in the following sub-sections.

A. OBU INITIAL SETUP

It is essential to verify the basic operations and configurations of the OBU before installing it in a vehicle. As a primary step, the firmware installed in the OBU was tested, and the firmware version was verified. This test was done to ascertain whether all the developed OBUs employ the same firmware version to avoid any compatibility issues. The firmware ver-

sion was tested using the ⟨VER⟩ keyset present in the OBU. By pressing the ⟨VER⟩ keyset, the current firmware version installed in the OBU is shown on the OBU display. After verifying the firmware version, the OBU IDs were tested for uniqueness. Each OBU should be given a unique ID for reference purposes in the RG algorithm, and the ID values ranged from 000 to 999. Initially, the ID was configured in each OBU by using the command ⟨SIOBDID, Name000⟩ given via serial port (from the computer to OBU) using hyper terminal software. After configuring, the ID was verified by the command ⟨GIOBDID⟩ using the hyper terminal.

Once the unique ID was configured, the GPS module and the wireless communication module connections were verified. These modules were connected to the central processing unit using serial port pins, and successful connections were verified with the individual LED indicators. Upon successful connection, the corresponding LED lights blink. After performing these initial test cases, each OBU was tested for its consistency. For this test, the OBU was switched ON, and all the essential information fed into the OBU would be displayed. The OBU displayed all the necessary information about the developer, firmware version, and the unique ID, indicating its readiness to perform the RG algorithm.

B. DATA UPLOAD AND GEO-FENCE RANGE SETUP

This section describes how the network information file uploading and the geo-fence configuration were tested. To test the successful uploading of the network file, a text file containing the details of the nodes (intersections), links (lanes), and the possible routes with predefined origin-destination (OD) pairs was created and uploaded to the OBU from the computer via serial port using the hyper terminal. The uploading process was initiated using the OBU ⟨DATA⟩ keyset. Upon the successful completion of the data transfer, the uploaded file was verified by the message ''Upload Completed'' on the OBU display. Also, using the ⟨SELECT⟩ keyset, the accuracy of the uploaded dataset was verified. From the uploaded file, the user can select the origin-destination pair with a predefined route to start their trip.

Also, the boundaries of the geo-fence range were verified. Initially, the default geo-fence range was configured as 200 m in every OBU, and it can be updated to any value using the ⟨SET-UP⟩ keyset in the OBU. The accuracy of the geo-fence range in every OBU was verified from the setup geo-fence screen option in the display unit.

C. OBU INFORMATION EXCHANGE

The OBUs should perform message exchange (send and receive) when they are within the predefined geo-fence range. To verify the message exchange operations, an OBU active trip was verified by the display of an 'TRIP ACTIVE' message to ensure that the vehicle was active on its trip. When the OBUs are within the geo-fence boundary, they should display an 'IN GEO-FENCE' message. Once the OBUs were present within the geo-fence, they transferred information according to the proposed RG algorithm standards. This was verified by the indication of the blinking RF LED present in the sender (candidate) OBUs. When an OBU received information from the candidate OBUs within the geo-fence region, it was verified by the indication of the blinking geo-fence LED present in the receiver (searcher) OBU.

D. OBU ROUTE UPDATE AND DATA DOWNLOAD

In this section, we verify the route update during the trip active time and trace file downloading after the trip time is tested in the OBU. Whenever a searcher OBU receives route update information from the opposite traveling candidate OBUs, the searcher OBU analyzes the received path information and selects the shortest route to travel. If the received path information to the destination has less travel time than the remaining predefined path travel time, the searcher OBU updates its path to the received path. The update on the searcher OBU can be verified by the indication of the OBU display message ($'U', 'U1', \ldots, 'Un'$), which is based on the number of updates it has received. When the searcher OBU does not update any path and follows the predefined remaining path, it is verified by the message indication 'A', which means that the OBU follows only the predefined path.

Once the OBU completes the trip, the trace file can be downloaded from the OBU for further analysis. Using the ⟨DATA⟩ keyset, the trace file of the OBU can be transferred from the OBU to the computer using the hyper terminal. The downloading operation can be performed only when the OBU ends its trip or when it is in its idle state. The successful downloading operation can be verified by the message 'DOWNLOADING' on the OBU display. If attempting to download the trace file during the trip active time, the operation gets aborted and the message 'TRIP ACTIVE' is displayed on the OBU.

Overall, six OBUs were fully developed, and their working conditions were verified with all functional tests. As the tests were performed on the developed OBUs individually, every OBU performed the test actions successfully without errors and functioned explicitly as mentioned in the above test cases. Further, the OBUs were utilized in a test-bed environment under different scenarios, as discussed in the next section.

VII. TEST-BED SCENARIOS

Upon the successful verification of the functionalities, as indicated earlier in Section VI, the OBUs were further tested in different test-bed scenarios in a controlled environment to reflect various operating conditions that may occur in a real-world environment. The purpose of these test-bed scenarios is to verify the RG algorithm's effectiveness in different situations by eliminating real-world factors created from communications that could be encountered during large-scale deployment such as interference. All the scenarios discussed in this paper were crafted carefully to show that the algorithm can effectively provide RG to vehicles even in the worst cases of deficient data exchange. In these following test cases, the focus is only on the RG update with respect

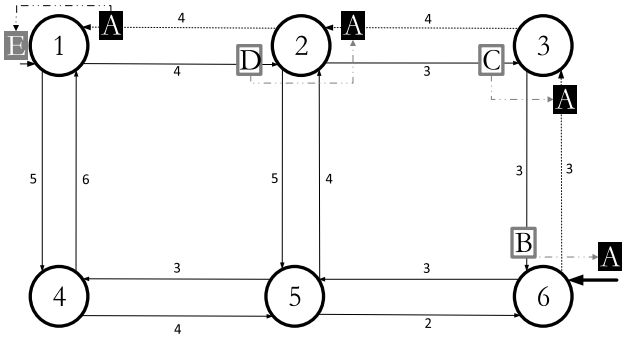


FIGURE 4. Network graph for the ideal test case with all possible link information.

to the searcher vehicle and to verify that the shortest path of the searcher vehicle to the destination is achieved. Other real-world compositions of road networks, link-speeds, link-lengths, and different levels of network congestion were used in carrying out real-life field tests in Al Ain, UAE. These field tests and results are not reported here and shall be discussed in another forthcoming article. To carry out the test-bed scenarios in the laboratory, a simplified network graph was created with nodes (intersections) connected by fixed travel time links. The geo-fence range for communication was set to 200 m for every node. A total of four different scenarios were created and tested, as discussed in the following sub-sections.

A. SCENARIO I: SEARCHER WITH ALL POSSIBLE LINK INFORMATION

In this scenario, an ideal condition was assumed for the RG algorithm, where the vehicle (represented here by OBU A) receives necessary information from the network at every geo-fence (G_i). The network, as shown in Fig. 4, was created in the test-bed environment. In this network, OBU A was set to be a moving vehicle with a predefined path $(6, 3), (3, 2), (2, 1)$ from origin node (6) to destination node (1). The other OBUs (B, C, and D) were set as "stationary" at the incoming links of the corresponding geo-fence regions opposite to the traveling direction of OBU A, as shown in Fig. 4. Each stationary OBU was configured with a clock timer such that it transmits the current link travel time to the searcher OBU. Also, the origin and destination nodes of these stationary OBUs were intentionally set to high values (999) such that their respective origins and destinations do not exist in the network (i.e., $D(v) \neq O(v')$). Finally, OBU E representing a searcher vehicle, where D (OBU E) $=$ O (OBU A), was configured with an initial assigned path of $(1, 2), (2, 5), (5, 6)$ and was kept waiting at node (1) within the geo-fence of (G_1) until it met the candidate OBU A to perform the message exchange.

The results of this configured scenario are summarized in Table 3. The test was initiated by OBU A starting to move from node (6) travelling along the predefined path to destination node (1). At every geo-fence region of nodes 6, 3, and 2 (intersections), OBU A received information from the stationary OBUs (B, C, and D). As a searcher, OBU A stored only the current link travel time information (i.e., recorded time because $D(v) \neq O(v')$) from the received information and used it to compute the reverse trajectory. Subsequently, when OBU A reached the geo-fence range of node (1) (G_1), it encountered OBU E and performed message exchange. With respect to OBU E, as its condition matched (i.e., D (OBU E) $=$ O (OBU A)) with OBU A, the searcher OBU E stored all the information from the candidate OBU A, as discussed in Section III.

From the received information, OBUE compared the received reversed trajectory path time (10 *min*) with the remaining predefined path time (12 *min*) and selected the shortest path to travel. In this test case, the OBUE updated the originally pre-defined route to the reversed trajectory path (1, 2), (2, 3), (3, 6) and started the trip to the destination. In this scenario, the proposed RG algorithm enabled OBU E to foresee the network before starting its trip and helped to capture the best route to travel. As the information exchange was carried out by the opposite travelling vehicles, any potential recent incident delays could have been effectively captured and reflected in the route update without any latency issues, which are problematic in many centralized systems.

B. SCENARIO II: SEARCHER WITH MISSING LINK INFORMATION

In this scenario, we tested the system when a vehicle (***OBU A***) does not find opposite travelling vehicles (i.e., ***OBU*** absent) at some geo-fence regions. As per the RG algorithm, when a searcher vehicle does not encounter any opposite vehicle along the incoming links of geo-fence ($\boldsymbol{G_i}$), it adds its own current link travel time to the summation of the travel time along the reversed trajectory path. That is, the travel time of the reversed link direction is assumed to be equivalent to the travel time of ***OBU A***. In this scenario, we adapted the test-bed, as shown in Fig. 5. This test is similar to Scenario I (Fig. 4) with ***OBU C*** at node (3) and ***OBU D*** at node (2) removed.

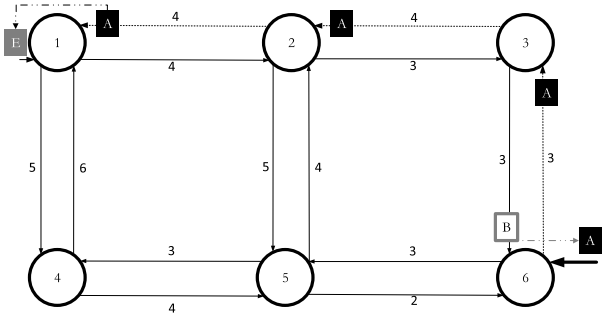


FIGURE 5. Network graph for the missing link information test case.

The test report is given in Table 4. When the test began, OBU A started its trip from the origin node (6) and moved along in its predefined path to the destination node (1). At the geo-fence range (G_6), OBU A encountered OBU B

TABLE 3. Test results of the ideal all possible link information scenario.

OBU	Origin	Destination	Moving (M) or Fixed (F)	Predefined path & Travel Time		Geo-fence Location (G_N) and Time (mins)	V2V Communication				Reverse Trajectory path & Travel Time	
							Opposite OBU	Algorithm Matching Condition (Yes / No)	Received Information			
				Path	Travel Time (mins)				Link/Path	Travel Time (mins)	Path	Total Travel Time (mins)
A	6	1	M	[6,3, 2,1]	(11)	$G_6, 0$	*B*	No	[3,6]	(3)	[3,6]	3
						$G_3, 3$	*C*	No	[2,3]	(3)	[2,3,6]	3+3=6
						$G_2, 7$	*D*	No	[1,2]	(4)	[1,2,3,6]	3+3+4=10
						$G_1, 11$	*E*	Yes	-	(0)	-	-
B	999	999	F	-	-	$G_6, 0$	*A*	No	-	(0)	-	-
C	999	999	F	-	-	$G_3, 3$	*A*	No	[6,3]	(3)	-	-
D	999	999	F	-	-	$G_2, 7$	*A*	No	[3,2]	(3)	-	-
E	1	6	F	[1,2, 5,6]	(12)	$G_1, 11$	*A*	Yes	[2,1]	(4)	-	-
									[1,2] + [2,3] + [3,6]	(10)	-	-

TABLE 4. Test results of the missing link information scenario.

OBU	Origin	Destination	Moving (M) or Fixed (F)	Predefined path & Travel Time		Geo-fence Location (G_N) and Time (mins)	V2V Communication				Reverse Trajectory path & Travel Time	
							Opposite OBU	Algorithm Matching Condition (Yes / No)	Received Information			
				Path	Travel Time (mins)				Path	Travel Time (mins)	Path	Total Travel Time (mins)
A	6	1	M	[6,3, 2,1]	(11)	$G_6, 0$	*B*	No	[3,6]	(3)	[3,6]	3
						$G_3, 3$	–	-	-	-	[2,3,6]	3+4=7
						$G_2, 7$	–	-	-	-	[1,2,3, 6]	3+4+4 =11
						$G_1, 11$	*E*	Yes	-	(0)	-	-
B	999	999	F	-	-	$G_6, 0$	*A*	No	-	(0)	-	-
E	1	6	F	[1,2, 5,6]	(12)	$G_1, 11$	*A*	Yes	[2,1]	(4)	-	-
									[2,1] + [3,2] + [3,6]	(11)	-	-

at the incoming links of the geo-fence range and received its information. As the origin destination pair did not match (D (OBU A) $\neq$ O (OBU B)), the searcher OBU A stored only OBU B's current link travel time information and computed the reverse trajectory path. Later, when OBU A moved to geo-fences (G_3) and (G_2), it did not encounter any opposite OBUs. Therefore, while leaving the geo-fence zones, as a searcher, OBU A added its own current link travel time to the summation of the travel time along its reversed trajectory path.

Similarly, when OBU A reached geo-fence (G_1), it encountered OBU E and exchanged information. From this information, the searcher OBU E compared the received reversed trajectory path time (11 *min*) from OBU A along with the remaining predefined path time (12 *min*) and selected the shortest path to travel. In this test case, OBU E updated the route to the received reversed trajectory path (1, 2), (2, 3), (3, 6)and started its trip to the destination.

In this scenario, the proposed RG algorithm estimates the travel time of the link when a searcher vehicle does not encounter opposite traveling vehicles. In real life, this scenario might be encountered in free-flow traffic conditions. If so, the searcher vehicle uses its own travel time to estimate the reversed trajectory path's travel time.

C. SCENARIO III: SEARCHER WITH MULTIPLE LINK INFORMATION

In this scenario, we test when a vehicle (OBU A) receives information from multiple vehicles (i.e., more than one OBU is in the geo-fence region (G_i)). As per the RG algorithm, when the searcher vehicle receives information from more than one opposite candidate vehicle, the searcher vehicle should update and store the most recent received information.

TABLE 5. Test results of the multiple link information scenario.

OBU	Origin	Destination	Moving (M) or Fixed (F)	Predefined path & Travel Time		Geo-fence Location (G_N) and Time (mins)	V2V Communication				Reverse Trajectory path & Travel Time	
				Path	Travel Time (mins)		Opposite OBU	Algorithm Matching Condition (Yes / No)	Received Information		Path	Total Travel Time (mins)
									Path	Travel Time (mins)		
A	6	1	M	[6,3,2,1]	(11)	$G_6, 0$	B	No	[3,6]	(3)	[3,6]	3
						$G_3, 3$	C	No	[2,3]	(3)	[2,3,6]	3+3=6
						$G_3, 4$	F	No	[2,3]	(2)	[2,3,6]	3+2=5
						$G_2, 7$	D	No	[1,2]	(4)	[1,2,3,6]	3+2+4=9
						$G_1, 11$	E	Yes	-	(0)	-	-
B	999	999	F	-	-	$G_6, 0$	A	No	-	(0)	-	-
C	999	999	F	-	-	$G_3, 3$	A	No	[6,3]	(3)	-	-
F	999	999	F	-	-	$G_3, 4$	A	No	[6,3]	(3)	-	-
D	999	999	F	-	-	$G_2, 7$	A	No	[3,2]	(4)	-	-
E	1	6	F	[1,2,5,6]	(12)	$G_1, 11$	A	Yes	[2,1]	(4)	-	-
									[1,2] + [2,3] + [3, 6]	(9)	-	-

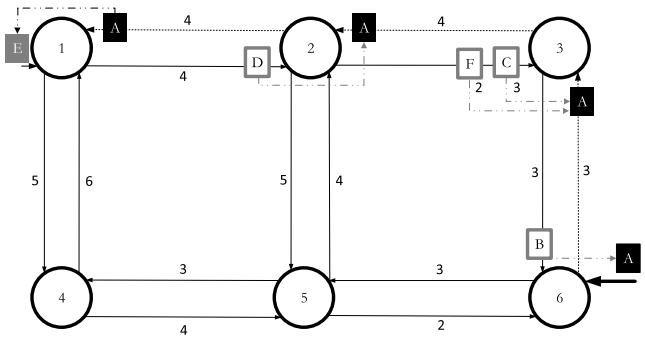


FIGURE 6. Network graph for the multiple link information test case.

For this scenario, we adapted the previous ideal test-bed scenario (in Fig. 4) with OBUs C and F at node 3, as shown in Fig. 6. OBUs C and F were introduced in the network at different times (3 and 4 minutes, respectively).

The results report is given in Table 5. OBU A started its trip from the origin node (6) and moved along in its predefined path to the destination node (1).

When OBU A moved to geo-fence (G_3), it encountered OBU C at time 3 and received information. As the searcher, OBU A stored only the current link information from the received information due to the condition mismatch D (OBU A) $\neq$ O (OBU C). Similarly, later at time 4, OBU A encountered OBU F at the same geo-fence (G_3) and received information. As the current link information from OBU F was more recent than that from OBU C, the searcher OBUA replaced the previously received OBU C information with the more recent information from OBU F and updated the reversed trajectory path.

Later, at time 7, OBU A at geo-fence range (G_2) encountered an opposite OBU D. As a searcher, OBU A stored only the current link travel time information exchanged by stationary OBU D due to the condition mismatch D (OBU A) $\neq$ O (OBU D) and OBU A added the received information to the summation of the reverse trajectory path.

Similarly, when OBU A reached geo-fence (G_1), it encountered OBU E and exchanged information. The searcher OBU E compared the received reversed trajectory path time(9 *mins*) with its remaining predefined path time (12 *mins*) and selected the shortest path to travel. In this case, OBU E updated the route to the reversed trajectory path (1, 2), (2, 3), (3, 6)and started its trip to its destination.

D. SCENARIO IV: BEHAVING SIMULTANEOUSLY AS A SEARCHER AND CANDIDATE VEHICLE

In this testing scenario, we test the RG algorithm when two OBUs of two vehicles, v and v, satisfy the conditions $D(v) = O(v)$ and $D(v\prime) = O(v)$. In such a case, each vehicle would act as a searcher and candidate vehicle simultaneously. As shown in Fig. 7, the network was created in the test-bed environment with (9) nodes. OBUs A and B were set to be the moving in opposite directions, where OBU A started from origin node (1) with a predefined path of (1, 4), (4, 5), (5, 8), (8, 9)to destination node (9), while OBU B started from node (9) with a predefined path of (9, 6), (6, 5), (5, 2), (2, 1) to destination node (1). As shown in the figure, the other OBUs (C, D, E and F) were configured with a clock timer, which transmits their current link travel times to the searcher OBUs. The OBUs (C, D, E and F) were placed as stationary along the incoming links of the corresponding geo-fence regions opposite to the traveling direction of OBU A and OBU B, respectively.

Additionally, the origin and destination nodes of these stationary OBUs were intentionally set to 999 such that the

TABLE 6. Test results for the behaving simultaneously as a searcher and candidate vehicle scenario.

OBU	Origin	Destination	Moving (M) or Fixed (F)	Predefined path & Travel Time		Geo-fence Location (G_N) and Time (mins)	V2V Communication				Reverse Trajectory path & Travel Time	
							Opposite OBU	Algorithm Matching Condition (Yes / No)	Received Information			
				Path	Travel Time (mins)				Path	Travel Time (mins)	Path	Total Travel Time (mins)
A	1	9	M	[1,4,5,8,9]	(16)	$G_1, 0$	*C*	No	[4,1]	(6)	[1,4]	6
						$G_4, 5$	*E*	No	[5,4]	(3)	[1,4,5]	6+3=9
						$G_5, 9$	*B*	Yes	[6,5]	(4)	[1,4,5,6,9]	6+3+2+2=13
									[5,6] + [6,9]	(4)		
B	9	1	M	[9,6,5,2,1]	(14)	$G_9, 0$	*D*	No	[6,9]	(2)	[9,6]	2
						$G_6, 5$	*F*	No	[5,6]	(2)	[9,6,5]	2+2=4
						$G_5, 9$	*A*	Yes	[4,5]	(4)	[9,6,5,4,1]	2+2+3+6=13
									[4,1] + [5,4]	(9)		
C	999	999	F	-	-	$G_1, 0$	*A*	No	-	(0)	-	-
D	999	999	F	-	-	$G_9, 0$	*B*	No	-	(0)	-	-
E	999	999	F	-	-	$G_4, 5$	*A*	No	[1,4]	(5)	-	-
F	999	999	F	-	-	$G_6, 5$	*B*	No	[9,6]	(5)	-	-

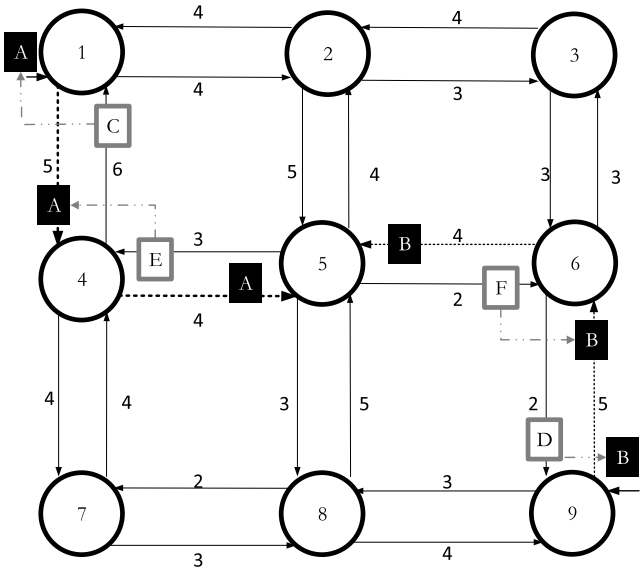


FIGURE 7. Network graph for the behaving simultaneously as a searcher and candidate vehicle test case.

algorithm matching condition always fails. The test scenario was planned such that both OBUs *A* and *B* move along the predefined path and meet at the common intermediate node (5). The test results report is given in Table 6. OBU *A* and OBU *B* started their trips from their respective sources to their destination nodes. At every node, OBU *A* and OBU *B* received information from the opposite stationary candidate OBUs and stored only the current link travel time information due to the condition mismatch $D(v) \neq O(v')$.

Finally, OBU *A* and OBU *B* were queued within geo-fence (G_5) at the same time, where they exchanged information. As both the origin and destination matched with each other (D(OBU *A*) = O(OBU *B*), D(OBU*B*) = O(OBU *A*)), both OBUs *A* and *B* acted as searcher and candidate simultaneously. Both the OBUs exchanged their reversed trajectory path and estimated travel time, as discussed in Section III. The received information by each OBU was compared with their own remaining predefined path times to select the shortest path to travel.

In this case, OBU *A* compared the received reversed trajectory path time (4 *mins*) with the remaining predefined path time (7 *mins*) and selected the reversed trajectory path [5, 6, 9] to the destination. On the other hand, OBU *B* compared the received reversed trajectory path time (9 *mins*) with the remaining predefined path time (8 *mins*) and selected the remaining predefined path [5, 2, 1] to destination. By exchanging information in both ways using inter-vehicular communication, the vehicles efficiently managed RG by themselves without any infrastructure support.

VIII. REAL-TIME FIELD TESTING

To test the technology in a real-life environment, a selected portion of the urban network of Al Ain city was selected, as shown in Fig. 8. The network has several intersections either operated by traffic signals or roundabouts. To configure the field tests, each test was registered as a network graph in the developed OBU ; the intersections and roundabouts were marked as nodes, and the connecting highways between any two successive nodes were considered as links. All nodes were coded using GPS coordinates.

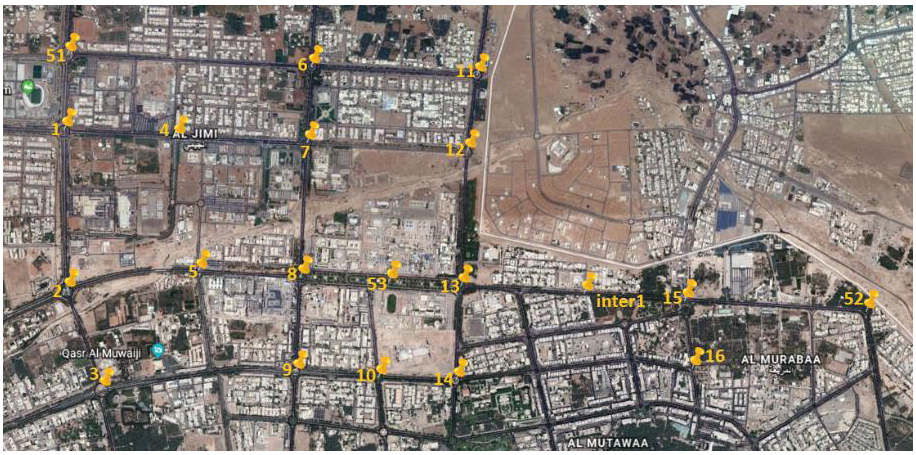


FIGURE 8. Field test network map (Al Ain city, UAE).

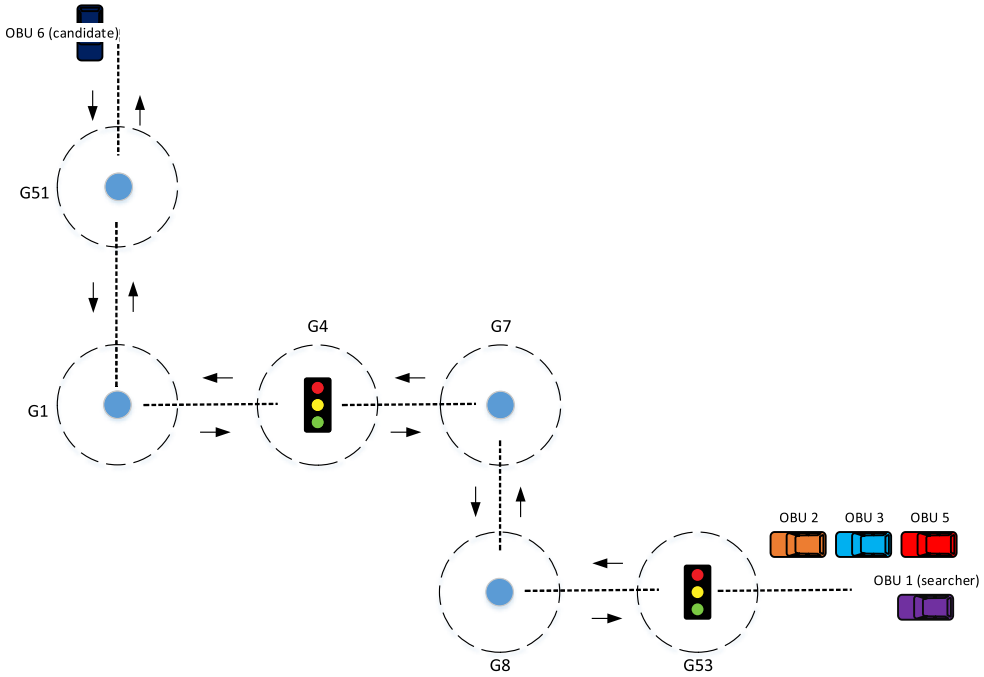


FIGURE 9. Route intersections and OBUs used in field test.

Due to space limitations, we shall limit our discussion here to one field test only. Other field tests will be discussed in other papers. In this test, geo-fences were centered at network intersection nodes (signalized intersections or roundabouts) and their radii were set to 200 m.

Five OBUs were installed in conventional vehicles: four travelling in one direction, and the fifth travelling in the opposite direction (Fig. 9). Three vehicles (OBU 2, 3, and 5) commenced their trips from origin node 53 to destination node 51 with a predefined path of 53−8−7−4−1−51. A two-minute headway was introduced to synchronize the four travelling vehicles to meet the opposite traveling vehicle (OBU 6) at different geo-fence locations. OBU 6 started the trip in the opposite direction with its origin as node 51 and destination as node 53 with a predefined path of $51-1-4-7-8-53$. OBU 6 (as a searcher) met the opposite candidate vehicles OBU 2, OBU 3, and OBU 5 at $G51$, $G1$, and $G8$, respectively. The received information by OBU 6 (as a searcher) is shown in Table 7. At $G53$, OBU 6 (as a candidate) met OBU 1 (as a searcher), and OBU 6 transmitted information, which was used by OBU 1 to select its best route (in comparison with its predefined path) to node 51.

Specific tests have been conducted for evaluating the performance of the V2V technology against Google Maps. These tests were conducted at three different times to consider various traffic conditions. The average length of the travelled routes was 5 km with link speeds ranging from 60 to 80 km/hr. The routes consisted of 5 nodes on average. The test results indicated that the difference between Google Maps and the true travel times could range from 1 to 30%. Google Maps overestimated the actual travel times in 78% of the studied cases and underestimated the actual values in 22% of the

TABLE 7. Information received by obu 6 as a searcher from candidate vehicles.

Candidate vehicle (v') *encountered by the searcher vehicle* $((v) = OBU6)$ *at geofence* (G_i)			*Information Transfered from candidate vehicle* (v') *to searcher vehicle* $((v) = OBU6)$		
Geofence (G_i)	*Candidate Vehicle ID* v'	*Current Link* (j,i) *of Candidate Vehicle at* (G_i)	*Received Current Link Travel Time* $t^r_{v'}(j,i)$ *in secs*	*Searcher Remaining Path* $(P^-(v')^r)$	*Received Travel Time on Remaining Path* $T_G(P^-(v')^r)$ *in secs*
G_{51}	OBU 2	$(1,51)$	40	$51-1-4-7-8-53$	417
G_1	OBU 3	$(4,1)$	32	$1-4-7-8-53$	249
G_4	–	$(7,4)$	85*	$4-7-8-53$	–
G_7	–	$(8,7)$	236*	$7-8-53$	–
G_8	OBU 5	$(53,8)$	32	$8-53$	32
G_{53}	OBU 1	(53_{Origin})	0	–	0

* The travel time information of OBU 6 was recorded since no candidate vehicles were encountered at the respective geo-fence locations.

cases. The average difference between Google Maps and the actual travel times was 14.82%.

The presented solution can be easily installed in any vehicle. It uses V2V communication, and as such, it alleviates all potential problems of complete network failure. Also, the solution can be easily integrated into any autonomous or conventional vehicle. In fact, it is built as a stand-alone system, and all it requires is a power connection to a vehicle battery.

IX. CONCLUSION

In this paper, the technology behind the development of the integrated embedded technical system for the real-time deployment of the RG algorithm by Hawas and El-Sayed [10] (OBU) and its laboratory test results were discussed in detail. For designing the OBU with specific hardware and software configurations, detailed functional system analysis was performed, and the technical system requirements were identified based on the input, processing, and output algorithm standards.

From the requirement analyses, the system design was carried out, and OBUs were built with reliable hardware, firmware, and database configurations. Six OBUs were integrated with a reliable processing module, GPS module, and communication module. Such OBUs can be easily installed in conventional vehicles to perform real-time RG.

To validate the efficiency and working conditions of each OBU, various tests were performed, and the OBUs' functionalities were tested individually. All the test results proved that the developed OBUs successfully perform their intended functionalities as per the RG algorithm standards.

To characterize the real-time behavior of the OBUs in different real-time hypothetical situations and network configurations, four different real-life scenarios were emulated in a lab environment, and the working model of the OBUs was tested by employing the RG algorithm. The test results showed that searcher vehicles were updated with the least travel time routes under the restricted geo-fence communication range at intersections.

The presented solution of V2V-based RG has many advantages, including less computation complexity, processing time, and required bandwidth compared to fixed infrastructure or centralized systems. As the exchanged information size is very small, and the system is efficient in sharing the information in restricted boundaries, the scheme can be integrated with any existing message standards that are currently used for creating vehicle safety cooperativeness [32], [33]. Based on the promising results obtained from the test-bed scenarios, the proposed system was deployed on real-world roadways and tested in two different urban cities (one in India and one in UAE). The results and analysis of other real-life deployment tests are to be discussed in future research papers.

Future research will include more detailed comparative analyses of the presented system with existing centralized systems such as Google and TomTom. Furthermore, varying the geo-fence dimensions has implications on communication requirements, possibility of data overflow, and solution effectiveness. This requires detailed study to capture the effect of varying the geo-fence range and to identify its optimal compromised value in light of the underlying network congestion levels.

ACKNOWLEDGMENT

The authors acknowledge Tata Consultancy Services (TCS) for the assistance they provided in developing the technology presented in this paper.

REFERENCES

[1] Y. Hawas and H. Mahmassani, "Comparative analysis of robustness of centralized and distributed network route control systems in incident situations," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1, no. 1537, pp. 83–90, 1996.

[2] H. S. Mahmassani and Y. E. Hawas, "Real-time dynamic traffic assignment for route guidance: Comparison of global predictive vs. Local reactive strategies under stochastic demands," *IFAC Proc. Volumes*, vol. 30, no. 8, pp. 543–548, 1997.

[3] Y. E. Hawas, "A cooperative distributed system for real-time route guidance," *J. Transp. Technol.*, vol. 2, no. 3, pp. 230–240, 2012.

[4] H. S. Mahmassani and S. Peeta, "System optimal dynamic assignment for electronic route guidance in a congested traffic network," in *Urban Traffic Networks*. Berlin, Germany: Springer, 1995, pp. 3–37.

[5] J. L. Adler, G. Satapathy, V. Manikonda, B. Bowles, and V. J. Blue, "A multi-agent approach to cooperative traffic management and route guidance," *Transp. Res. B, Methodol.*, vol. 39, no. 4, pp. 297–318, 2005.

[6] L. Chen, C. Jiang, and J. Li, "VGITS: ITS based on intervehicle communication networks and grid technology," *J. Netw. Comput. Appl.*, vol. 31, no. 3, pp. 285–302, 2008.

[7] J. Pan, I. S. Popa, K. Zeitouni, and C. Borcea, "Proactive vehicular traffic rerouting for lower travel time," *IEEE Trans. Veh. Technol.*, vol. 62, no. 8, pp. 3551–3568, Oct. 2013.
[8] Y. E. Hawas, "A microscopic simulation model for incident modeling in urban networks," *Transp. Planning Technol.*, vol. 30, nos. 2–3, pp. 289–309, 2007.
[9] J. Miller, "Vehicle-to-vehicle-to-infrastructure (V2V2I) intelligent transportation system architecture," in *Proc. IEEE Intell. Vehicles Symp.*, vol. 4, Jun. 2008, pp. 715–720,
[10] Y. E. Hawas and H. El-Sayed, "Autonomous real time route guidance in inter-vehicular communication urban networks," *Veh. Commun.*, vol. 2, no. 1, pp. 36–46, 2015.
[11] R. Claes, T. Holvoet, and D. Weyns, "A decentralized approach for anticipatory vehicle routing using delegate multiagent systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 364–373, Jun. 2011.
[12] Google Maps Research. (2017). *Distance & Duration for Multiple Destinations*. Accessed: Oct. 20, 2017. [Online]. Available: https://developers.google.com/maps/documentation/distance-matrix/
[13] (2017). *TomTom MyDrive*. Accessed: Oct. 22, 2017. [Online]. Available: https://mydrive.tomtom.com/en_gb
[14] N. Gupta, A. Prakash, and R. Tripathi, "Medium access control protocols for safety applications in vehicular ad-hoc network: A classification and comprehensive survey," *Veh. Commun.*, vol. 2, no. 4, pp. 223–237, Oct. 2015.
[15] H. Vahdat-Nejad, A. Ramazani, T. Mohammadi, and W. A. Mansoor, "A survey on context-aware vehicular network applications," *Veh. Commun.*, vol. 3, pp. 43–57, Jan. 2016.
[16] J. A. F. F. Dias, J. J. P. C. Rodrigues, and L. Zhou, "Cooperation advances on vehicular communications: A survey," *Veh. Commun.*, vol. 1, no. 1, pp. 22–32, 2014.
[17] S. Boussoufa-Lahlah, F. Semchedine, and L. Bouallouche-Medjkoune, "Geographic routing protocols for vehicular ad hoc networks (VANETs): A survey," *Veh. Commun.*, vol. 11, pp. 20–31, Jan. 2018.
[18] R. S. Bali, N. Kumar, and J. J. P. C. Rodrigues, "Clustering in Vehicular ad hoc networks: Taxonomy, challenges and solutions," *Veh. Commun.*, vol. 1, no. 3, pp. 134–152, Jul. 2014.
[19] M. Amoozadeh, H. Deng, C.-N. Chuah, H. M. Zhang, and D. Ghosal, "Platoon management with cooperative adaptive cruise control enabled by VANET," *Veh. Commun.*, vol. 2, no. 2, pp. 110–123, Apr. 2015.
[20] M. Chaqfeh, A. Lakas, and I. Jawhar, "A survey on data dissemination in vehicular ad hoc networks," *Veh. Commun.*, vol. 1, no. 4, pp. 214–225, Oct. 2014.
[21] Y. E. Hawas, M. J. B. Napeñas, and Y. Hamdouch, "Comparative assessment of intervehicular communication algorithms for real-time traffic route guidance," *J. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 199–217, 2009.
[22] K. Dar, M. Bakhouya, J. Gaber, M. Wack, and P. Lorenz, "Wireless communication technologies for ITS applications [topics in automotive networking]," *IEEE Commun. Mag.*, vol. 48, no. 5, pp. 156–162, May 2010.
[23] L. Li, D. Wen, and D. Yao, "A survey of traffic control with vehicular communications," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 425–432, Feb. 2014.
[24] M. B. Younes and A. Boukerche, "Intelligent traffic light controlling algorithms using vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 8, pp. 5887–5899, Aug. 2016.
[25] J.-W. Ding, C.-F. Wang, F.-H. Meng, and T.-Y. Wu, "Real-time vehicle route guidance using vehicle-to-vehicle communication," *IET Commun.*, vol. 4, no. 7, pp. 870–883, Apr. 2010.
[26] M. Wei and Y. Meng, "Research on the optimal route choice based on improved Dijkstra," in *Proc. IEEE Workshop Adv. Res. Technol. Ind. Appl. (WARTIA)*, Sep. 2014, pp. 303–306.
[27] S. Fujii *et al.*, "Cooperative vehicle positioning via V2V communications and onboard sensors," in *Proc. IEEE Veh. Technol. Conf. (VTC Fall)*, vol. 5, Sep. 2011, pp. 1–5.
[28] J. S. Pan, I. S. Popa, and C. Borcea, "Divert: A distributed vehicular traffic re-routing system for congestion avoidance," *IEEE Trans. Mobile Comput.*, vol. 16, no. 1, pp. 58–72, Jan. 2017.
[29] J. Lin, W. Yu, X. Yang, Q. Yang, X. Fu, and W. Zhao, "A real-time en-route route guidance decision scheme for transportation-based cyberphysical systems," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2551–2566, Mar. 2017.
[30] "Tools by ARM," Keil, Grasbrunn, Germany, White Paper, 2009. Accessed: Oct. 23, 2017. [Online]. Available: https://www.keil.com/product/brochures/uv4.pdf
[31] Flash Magic. (2017). *Application*. Accessed: Oct. 23, 2017. [Online]. Available: http:// www.flashmagictool.com/assets/resources/89LPC932AppNote.pdf
[32] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.
[33] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*, Standard ETSI EN 302 637-2 V1.3.1, Sep. 2014.

YASER E. HAWAS received the Ph.D. degree from The University of Texas at Austin, in 1996. He currently serves as a Professor with the Department of Civil and Environmental Engineering, UAE University, where he also serves as the Director of the Roadway, Transportation, and Traffic Safety Research Center. His primary areas of expertise include the development of intelligent transportation systems, advanced control systems, megaproject management, traffic network operation, and safety analysis. He was a recipient of the H. H. Shiekh Khalifa (the UAE President) Education Award in the Category of Distinguished University Professor in the Field of Scientific Research, for the year 2011–2012.

GOKULNATH THANDAVARAYAN received the Master of Computer Science Engineering degree from Anna University, India, in 2012. After completing his degree, he was a Lecturer in an Engineering College for three years, tutoring computer science courses. In 2015, he joined as a Research Associate in United Arab Emirates University, United Arab Emirates, and worked in different projects related to vehicular networks. His research interests include wireless networks, vehicular communication, and VANET applications.

BASIL BASHEERUDEEN received the master's degree in transportation engineering and management from the National Institute of Technology, Tiruchirappalli, India, in 2012. After completion, he served as a Lecturer tutoring traffic engineering and urban transport planning in an Engineering College in Kerala, for two years. Later, he joined as a Research Assistant with the Department of Civil Engineering, IIT Bombay, India, where he was involved in transport planning and urban traffic operation studies. In 2016, he joined as a Research Associate with the Roadway Transportation and Traffic Safety Research Centre, UAE University, Al Ain. His current research interests include intelligent transportation systems, connected vehicles, transport security, traffic simulation, and operations with an emphasis on urban traffic control.

MOHAMMAD SHERIF received the master's degree in computer applications from Anna University, Chennai, India, in 2009. He is currently a Research Associate and a Computer Programmer with the Roadway, Transportation, Traffic and Safety Research Center, United Arab Emirates University, Al Ain, United Arab Emirates. His research interests include intelligent transportation systems, connected vehicles, transport security, and project management. He has interest in developing web and desktop-based computer applications as well.