Leader and Leaderless Multi-layer Consensus with State Obfuscation: an Application to Distributed Speed Advisory Systems

Wynita Griggs, *Member, IEEE,* Giovanni Russo, *Member, IEEE,* and Robert Shorten, *Senior Member, IEEE*

***Abstract*—Two new distributed Speed Advisory Systems (SASs) are introduced in this paper. The systems implement consensus algorithms that guide a set of vehicles towards a common driving speed. A major innovation is that consensus is achieved over a multi-layer network, where parallel network topologies of connected vehicles are superimposed. The reason for the use of these parallel networks is that, in this way, state obfuscation is possible, with the benefit that common driving speed is attained with no vehicle knowing the exact state of other vehicles. Convergence of the SASs is formally proven and two new results for the consensus of multi-layer networks modelled via stochastic differential equations are introduced. The SASs are also validated via simulation and via a Hardware-in-the-Loop set-up, where a real vehicle interacts with simulated entities.**

I. Introduction

SINCE their original conception, the goal of Intelligent Speed Adaptation (ISA) systems has been to promote safe driving by alerting drivers when road speed limits are exceeded [2], driving at unsafe speeds being a significant issue worldwide [3], [4]. Since then, ISA systems have considerably evolved into *smarter* systems in an attempt to offer additional benefits to drivers and the surrounding environment. A remarkable example is given by speed advisory systems (SASs). The goal of such systems is to recommend suggested speeds to drivers, and they have proven useful in reducing general road traffic chaos and preventing travelling delays [5], [6]. The additional benefits of SASs include, but are not limited to: (i) ensuring that vehicles travel at safe speeds and at safe distances from the vehicles ahead of them [7]; (ii) maintaining, as much as possible, the free flow of traffic, navigating optimally through bottlenecks when they occur, thus increasing overall throughput on roads [6], [8]; (iii) helping to reduce factors such as polluting emissions and fuel consumption [5], [9], [10]; and safer driving in adverse conditions [11].

When designing SASs, it is typically assumed that the vehicles participating in this *service* exchange their state. However, recent studies show that, while individuals are willing to accept a plethora of new services like SASs, they are also reluctant to share their state with others [12]. An interesting problem in the design of intelligent transportation systems is therefore that of allowing vehicles to benefit from collaborative services without the need for them to fully disclose their state with the other vehicles in the system. In this context, we present here two SASs that guide vehicles to a common, desired, driving speed while, at same time, *obfuscating*, with some noise, the input received by each vehicle.

This work was partially supported by Science Foundation Ireland grant 11/PI/1177.

A simplified version of some of the results of this paper has been preliminarily presented at the 19th International IEEE Conference on Intelligent Transportation Systems [1].

W. Griggs and R. Shorten are with University College Dublin, School of Electrical, Electronic and Communications Engineering, Belfield, Dublin 4, Ireland. Email: wynita.griggs@ucd.ie, robert.shorten@ucd.ie

G. Russo is with IBM Research Ireland, Optimisation and Control Group, Dublin, Ireland. Email: grusso@ie.ibm.com

Related Work

Consensus problems have a long history in the field of computer science [13], where groups of agents have to agree upon certain quantities of interest. Due to its potential application in a number of fields, like data fusion and sensor networks, consensus has also been widely investigated within the control theoretical community; see, for example, [14]–[18]. Recently, consensus-based approaches have also been used to implement speed recommender systems; see, for example, [19], [20] and the references therein. In [21], for instance, eco-driving was examined in regards to controlling vehicles' speeds at a microscopic level to reduce fuel consumption and guarantee data delivery. Vehicles in the study were presumed to have V2V and V2I capabilities, and platoons were considered and treated as a single, large vehicle. In [22], the focus of the study was to extend the functionality of a traffic simulator and develop APIs for V2V and V2I communication. The extended simulation system was used to implement advisory speed recommendation and re-routing guidance for urban freeways under various load conditions. In [23], an application was proposed to reduce the CO_2 emissions of vehicles travelling along highways. A notion of optimised consensus was used to solve the emission optimisation problem and it was shown that total CO_2 emissions could be minimised if all vehicles followed the reference speed signal derived. A common factor of the solutions of the papers referenced above is that all of the vehicles share their speeds with other vehicles. In [9], an extended version of [23], a step towards preserving driver privacy was highlighted in that vehicles participating in the algorithm reported did not share their cost functions with other vehicles: only their recommended speeds.

Contributions of this Paper

In this paper, we propose two SASs with the objective of guiding a set of moving vehicles towards a common driving

speed. In contrast to previous algorithms, which strive to achieve consensus on a common driving speed via algorithmic innovation, one of the key novelties of our work is that it explores the benefits that can be achieved by allowing consensus protocols to evolve over multiple parallel network topologies. These parallel networks allow a way to *obfuscate*, via some noise, the input received by the vehicles. As a result of this, a single car in the network does not *see* the exact speed of its neighbours, but it rather sees a signal which is obfuscated by noise. To the best of our knowledge, our approach is new.

The algorithms we propose rely on a solid theoretical background. The theoretical results used to prove algorithmic convergence are also a novel contribution of the paper. Specifically, in the appendix, new sufficient conditions are given for the consensus in multi-layer networks, [24], modelled by a set of continuous-time stochastic differential equations. By using stochastic Lyapunov techniques [25], we introduce new sufficient conditions for network consensus. It is furthermore important to note that we are designing SASs rather than cooperative cruise control systems. This distinction is important. In a SAS, the speed that is delivered to the vehicle is a recommended speed for the driver and is not used to directly adjust the speed of the vehicle. As a consequence, as discussed in [9], string stability effects, which are a fundamental limitation of many cooperative control architectures [26]–[29], can be ignored in the design of a SAS. We remark that our focus in this paper is not to construct a fully decentralised system, but rather to construct a partially distributed solution which allows convergence of the vehicles towards a target speed, without requiring vehicles to directly exchange information related to their speed and their cost function.

Finally, another contribution of this paper is that we demonstrate our algorithms using a hardware-in-the-loop (HIL) platform, originally described in [30], which permits us to merge scenarios created using the microscopic traffic simulation package SUMO [31], capable of emulating large-scale traffic networks, together with proof-of-concept real-life vehicles equipped with the SASs that we propose in this work. That is, we "embed" a real vehicle, equipped with our SASs, into a SUMO simulation, this real vehicle being represented in the simulation visually by an avatar. Meanwhile, purely simulated vehicles participating in the same SUMO simulation are also subject to our SASs. Consequently, we provide a real driver the experience of participating in our speed advisory service in a large-scale scenario, where the rest of the participating traffic is simulated.

The rest of the paper is organised as follows. In Section II, the general architecture of our proposed SASs is presented. In Sections III and IV, we describe the base station and in-car algorithms as components within this architecture. In Section V, validations of the algorithms are performed using the HIL platform introduced above. Section VI offers a concluding summary of our work and avenues for future research. In the appendix, mathematical proofs of convergence relating to our algorithms are provided.

II. The Proposed Architecture and Goals

The general architecture of the SASs presented in this paper is schematically shown in Fig. 1. The key components of the architecture are: (i) a *base station* which controls a given geographic area; and (ii) an *in-car* system which establishes a connection with the base station and shows speed recommendations to drivers. The area controlled by the base station might be a stretch of road (e.g. a highway) or a *mini-city* (e.g. a university campus like the one used in Section V below for HIL testing).

The goal for the SASs presented in this paper is that of providing speed indications to a network of N interconnected vehicles in order to allow drivers in the same area to fulfill a common driving speed. In the first SAS we present, we assume that the only data sent from the i-th vehicle to the base station is the vehicle speed, v_i. In response, the base station returns to the i-th vehicle a distributed, obfuscated signal, u_i, which is elaborated on by the in-car system. The in-car system is then responsible for elaborating on u_i and for displaying to the i-th driver the recommended speed. If the indications are followed by drivers, then the SAS allows all the vehicles to achieve a common driving speed, which averages the speeds of all the vehicles in the network. In the rest of the paper, we will refer to this SAS as *leaderless*. Note that the speed towards which all the vehicles converge is not known a priori. This consideration motivated the second SAS we are presenting in this paper, which will be referred to as *SAS with leader* in what follows. In this case, the data set sent from the i-th vehicle to the base station includes both v_i and a cost function, h_i. The cost function quantifies the emissions of each vehicle while driving at a given speed. In response, the SAS again returns to the i-th vehicle an obfuscated signal u_i, which is then elaborated on by the in-car system. However, in this case, if the indications provided by the system are followed, then vehicles achieve a common driving speed which minimises the emissions. As

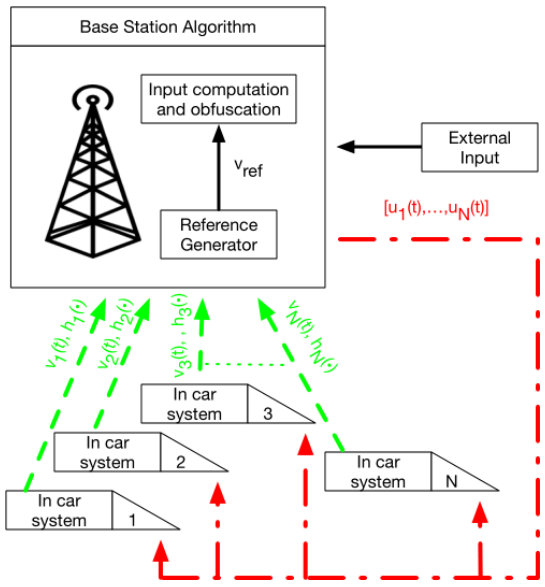

Fig. 1. Schematic diagram illustrating the system presented in this paper.

also shown in Fig. 1, this SAS is able to receive a reference speed from *external entities*. This feature has been included in order to handle scenarios where an external authority (e.g. police or city council) needs to broadcast a temporary speed restriction in the area controlled by the base station (due to e.g. weather conditions and/or road accidents).

Remark 1: While SASs make perfect sense in a highway scenario, they are also useful in a number of urban situations. For example, this is the case when a campus owner (such as an University) is interested in minimising the pollution footprint or energy consumption. The results presented in this paper are applicable in both urban and highway scenarios.

III. The Base Station Algorithm

In this section, we give the details of the base station algorithms for the SASs. Essentially, in both cases, the base station implements a consensus algorithm for a multi-layer network, with state obfuscation. We first present in Section III-A the consensus algorithm for the leaderless SAS. Then, in Section III-B, we present the algorithm for the SAS with leader.

For both SASs, the multi-layer network consists of two layers, with each layer having the number of nodes equal to N, i.e. equal to the number of vehicles within the area controlled by the base station. We also remark here that for both the leader and leaderless SAS, the signal received by each vehicle is *corrupted* by some white noise injected by the base station. This *obfuscation* implies that the i-th in-car system does not *see* the exact speed of the neighbouring vehicles and that, in this sense, the algorithm is privacy preserving. In the appendix, we will formally prove convergence of the algorithms and we will indeed show that the noise injected by the base station is crucial for the convergence.

Remark 2: Our approach, based on the use of multi-layer architectures, is motivated by certain situations of interest for Smart Cities and IoT applications, where several communication networks are available. For example, most car platforms now support 4G and there are also special V2V network protocols, as well as other dedicated I2V protocols. Street lighting and parked vehicles [32] are also being proposed as infrastructure to support V2V. In such situations, it makes sense to investigate how these networks can collaborate with each other to support advanced applications. In addition, in the context of stochastic differential equations, the dynamics that we consider naturally model networks of multiple layers.

Remark 3: The base station plays a key role for the algorithms presented in this paper. The basic assumption supporting the use of the architecture of Figure 1 is that the base station owner is honest and protects the data.

A. Leaderless SAS

This SAS only takes as input the speed of the vehicles within the area (i.e. the v_i's) and its goal is to guide the vehicles towards a common (average) driving speed. Essentially, such an algorithm continuously checks for vehicles entering/exiting the area controlled by the base station. This is done by creating a list that is updated: (i) every time a new vehicle entering the area is detected; (ii) whenever a vehicle exits from the controlled area. Once the list is updated, the algorithm gathers the speeds of the vehicles within the list and constructs the distributed input u_i by implementing the multi-layer network of Fig. 2.

Namely, the network consists of the following two layers.

- A first layer (i.e. the bottom layer in Fig. 2) that combines the speed of the last vehicle entering the area with the speed of the one that entered immediately before. This yields the computation of a *clean* input for the vehicles, i.e. u_i^c, $i = 1, \dots, N$, where

$$\begin{aligned} &u_i^c(t) := v_{i+1}(t) + v_{i-1}(t) - 2v_i(t),\ i = 2, \dots, N-1; \\ &u_1^c(t) := v_2(t) - v_1(t); \\ &u_N^c(t) := v_{N-1}(t) - v_N(t). \end{aligned} \tag{1}$$

- A second layer (i.e. the top layer in Fig. 2) that combines all the vehicle speeds and corrupts this aggregated information with some white noise, say $w(t)$ (i.e. noise is injected in this layer). The result of this computation is a *noisy* input for the vehicles, i.e. u_i^n, $i = 1, \dots, N$, where

$$u_i^n(t) := w(t) \sum_{j=1}^{N} (v_j - v_i). \tag{2}$$

- The two components are then summed and the recommended input that is returned to the i-th vehicle is

$$u_i(t) := u_i^c(t) + u_i^n(t). \tag{3}$$

We remark here that the noisy signal $w(t)$ in (2) is a gain for the signal $\sum_{j=1}^{N}(v_j - v_i)$. The key macro-steps of the base station algorithm are given in Algorithm 1. In the algorithms that follow, the lines **while** True **do** … **end while** indicate a standard infinite loop, where all of the operations between these lines are repeated.

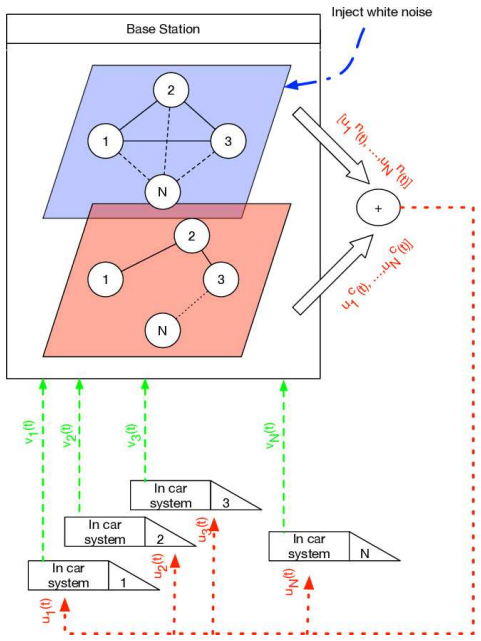

Fig. 2. Schematic diagram illustrating how the control input to the network of vehicles is computed for the leaderless SAS.

Algorithm 1 Leaderless SAS

function $[u_1, \dots, u_N]$ = LEADERLESS-SAS($[v_1, \dots, v_N]$)
  Internal Variables:
  List = list of vehicles within the controlled area
  while True **do**
    List update
    if new *vehicle* within the controlled area **then**
      append *vehicle* to *List*
    else if *vehicle* exits from the controlled area **then**
      remove *vehicle* from *List*
    end if
    Gather data
    $N \leftarrow$ size of *List*
    for *vehicles* in *List* **do**
      get speeds, v_i
    end for
    Generate noise
    $w \leftarrow$ value from white noise
    Implement the bottom and top network layers
    $u_1^c \leftarrow v_2 - v_1$
    $u_N^c \leftarrow v_{N-1} - v_N$
    for i in $[2, N-1]$ **do**
      $u_i^c \leftarrow v_{i-1} + v_{i+1} - 2v_i$
    end for
    for i in $[1, N]$ **do**
      $u_i^n \leftarrow w \cdot \sum_{j=1}^{N}(v_j - v_i)$
    end for
    Set output
    for i in $[1, N]$ **do**
      $u_i = u_i^n + u_i^c$
    end for
    return $[u_1, \dots, u_N]$
  end while
end function

B. SAS with Leader

This SAS takes into account the speeds of the vehicles, their cost functions (i.e. the h_i's) and (eventually) a reference speed provided by an external authority, say $\bar{v}$. These inputs are used by the base station to generate a reference driving speed, say v_{ref}, for the vehicles. The reference v_{ref} is generated as follows.

- If the cost functions from each single vehicle are available, then v_{ref} is computed by solving[1] the optimisation problem

$$\begin{aligned} &\min \textstyle\sum_{i=1}^{N} h_i(v_i), \\ &\text{s.t. } v_i = v_j, \ \forall i, j = 1 \dots N. \end{aligned} \tag{4}$$

In the context of this paper, each function $h_i(v_i)$ quantifies the vehicle emissions when travelling at speed v_i (such functions are consolidated in the literature, see e.g. [34], and are typically convex).

- In the case where $\bar{v}$ is given, then this overrides the optimisation problem and v_{ref} simply becomes the external reference (i.e. $v_{ref} = \bar{v}$).

Once v_{ref} is determined, then the distributed signals, u_i's, are constructed. Again, the u_i's are computed via a multi-layer network (see Fig. 3). Note that, in this case, nodes are

[1]In the validations of Section V, we employed the use of CVXPY, a Python-embedded modelling language for convex optimisation problems [33]. We remark here that our theoretical results are independent of the specific software being used to solve the optimisation problem.

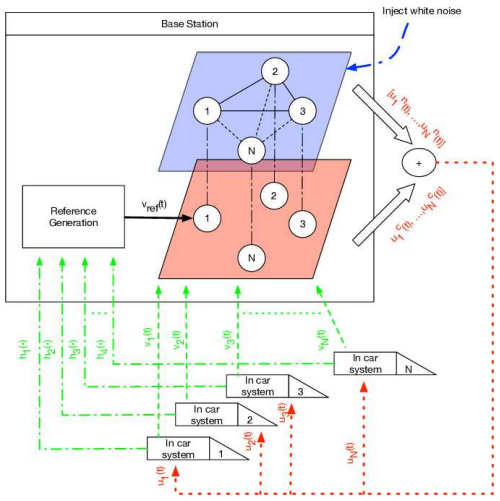

Fig. 3. Schematic diagram illustrating how the control input to the network of vehicles is computed for the SAS with leader.

disconnected at the bottom layer (i.e. the noise-free layer) and only one node receives v_{ref} as input (this node will act as network leader).[2] Nodes are instead connected through the top layer by means of noise injection. Namely, the network consists of the following two layers.

- The first layer (i.e. the bottom layer in Fig. 3) sends the reference speed, v_{ref} to a vehicle, say Vehicle 1. In the case where Vehicle 1 exits from the controlled area, then the reference is sent to the last vehicle entering the area. The clean input for the i-th vehicle is

$$\begin{aligned} u_1^c(t) &:= v_{ref}(t) - v_1(t); \\ u_i^c(t) &:= 0, \ \forall i = 2, \dots, N. \end{aligned} \tag{5}$$

- The second layer (i.e. the top layer in Fig. 3) computes u_i^n, $i = 1, \dots, N$, following (2).
- The two components are again summed up following (3) to obtain u_i.

The key steps of the base station algorithm are shown in Algorithm 2.

Remark 4: Algorithm 2 optimises a global cost function, while Algorithm 1 only ensures that the vehicles' speeds converge, without any guarantee of optimality for the consensus speed. That is, Algorithm 1 only ensures the vehicles' speeds converge while Algorithm 2 ensures that the speeds convergence towards an optimised value.

Remark 5: note that both of the SASs described in this paper make use of the actual speed of the vehicles within the area controlled by the base station. If the drivers follow the indications then, as shown in the appendix, convergence of the SASs is guaranteed. In principle, the SAS might take as input only the initial speeds of the vehicles when these enter the geographic area controlled by the base station. In this case, dynamics (1), (2), (3), (4) and (5) would be the dynamics for the recommended speed. The output u_i would be displayed only after the algorithm converges, i.e. $v_i = v_j$,

[2]This disconnected layer is a novel feature of the algorithm proposed here, in comparison to the work of [9]. Note that, in consensus literature, it is assumed that the network is connected.

$\forall i, j = 1, \ldots, N$. The convergence analysis in the appendix does not depend on whether the speeds in (1), (2), (3), (4) and (5) are "real" or recommended.

Remark 6: finally, note that, in (5), the reference speed $v_{ref}(t)$ is only given to one vehicle (namely, Vehicle 1). Note also that, for all the other vehicles, $u_i^c(t)$ is equal to 0. While this might seem surprising, this choice allows all the vehicles' speeds to converge towards the desired reference speed $v_{ref}(t)$. This is formally proven in the appendix.

Algorithm 2 SAS with leader

function $[u_1, \ldots, u_N]$ = LEADER-SAS($\bar{v}$,$[v_1, \ldots, v_N]$, $[h_1, \ldots, h_N]$)
  Internal Variables:
  List = list of vehicles within the controlled area
  while True **do**
    List update
    if new *vehicle* within the controlled area **then**
      append *vehicle* to *List*
    else if *vehicle* exits from the controlled area **then**
      remove *vehicle* from *List*
    end if
    Gather data
    $N \leftarrow$ size of *List*
    for *vehicles* in *List* **do**
      get speeds, v_i
      get cost functions, $h_i(v_i)$
      get external reference, $\bar{v}$
    end for
    if $\bar{v}$ available **then**
      $v_{ref} \leftarrow \bar{v}$
    else
      $v_{ref} \leftarrow$ solution optimisation problem (4)
    end if
    Generate noise
    $w \leftarrow$ value from white noise
    Implement the bottom and top network layers
    $u_1^c \leftarrow v_{ref} - v_1$
    for i in $[2, N]$ **do**
      $u_i^c \leftarrow 0$
    end for
    for i in $[1, N]$ **do**
      $u_i^n \leftarrow w \cdot \sum_{j=1}^{N}(v_j - v_i)$
    end for
    Set output
    for i in $[1, N]$ **do**
      $u_i = u_i^n + u_i^c$
    end for
    return $[u_1, \ldots, u_N]$
  end while
end function

IV. The In-Car System Algorithm

The in-car system is common to both of the SASs presented in this paper. The role of such a system is essentially that of facilitating the exchange of data with the base station and to provide the recommended speed to the driver. Specifically, whenever a connection is established, the vehicle speed is sent to the base station. Analogously, whenever $u_i(t)$ is received, this is converted to show the recommended speed to the driver. Note that the input received by each car $u_i(t)$ is physically an acceleration. The in-car system integrates $u_i(t)$ to get the recommended speed.

V. Validation

To demonstrate the efficacy of our proposed SASs, we considered the road network of a university campus (i.e. the University College Dublin, Belfield campus, in Ireland), and utilised the open source microscopic traffic simulation package, SUMO [31], to perform our emulations. The package comes with a *remote control interface*, TraCI [35], that allows one to adapt the simulation and to control individual vehicles on the fly.

A map of the University College Dublin (UCD) Belfield campus was imported from OpenStreetMap. The map was downloaded and edited using JOSM [36] and cleaned with XMLStarlet [37] before applying SUMO's *netconvert*. A maximum allowed link speed of 30km/h was set on the campus roads to reflect real-world speed limits, and a maximum allowed link speed of 50km/h was set on roads on the approach to the campus (i.e. on links on the approach to the campus entrance gates). The road network is shown in Fig. 4.

For traffic, we elected to simulate a scenario with passenger vehicles being allocated to enter one of three possible entrance gates and then heading to one of three possible car parks. These entrance gates and car parks are also depicted in Fig. 4. The vehicles' routes were chosen randomly from a total of nine possible routes. That is, each of the nine routes that any vehicle could potentially take were given equal weight, which thus formed the distribution from which the route for any particular vehicle was randomly selected from as it was added to the network. Such distributions can be provided to SUMO by adding the routeDistribution tag[3] to the scenario's route file. Vehicles entered the network at a rate of one vehicle per twenty seconds, for twenty minutes, at maximum permitted departure speed. By the conclusion of the simulation run, sixty vehicles in total had thus entered the network. When vehicles arrived at their destination car park, they were removed from the network. Other attributes[4] that we assigned to the simulated vehicles are as follows: max. speed = 70m/s; max. acceleration = 2.6m/s^2; max. deceleration = 4.5m/s^2; speedFactor = 1; speedDev = 0.1.

To assess the convergence of our algorithms, Python scripts were written which implemented Algorithms 1, 2 and the in-car system algorithm. In such scripts, the in-car algorithm receives the acceleration computed by the base station algorithm and converts it to a speed. In our simulations, two different test case scenarios were considered: (i) in the first case, no vehicle was instructed to be a leader and thus speeds were expected to converge towards the average (Algorithm 1); (ii) in the second case, the first vehicle to enter the simulation was selected as the leader (Algorithm 2) and when this vehicle exited from the simulation, the leadership was passed to another vehicle. The time step size used in all of the emulations was 0.1s. In cases (i) and (ii), the noise, w, injected by the base station algorithm was generated from a Gaussian distribution with mean equal to 0 and standard deviation equal to 0.5.

[3]See http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Route_and_vehicle_type_distributions. Last accessed: 9th January 2017.

[4]Attribute descriptions can be found in the user documentation on the SUMO website [38].

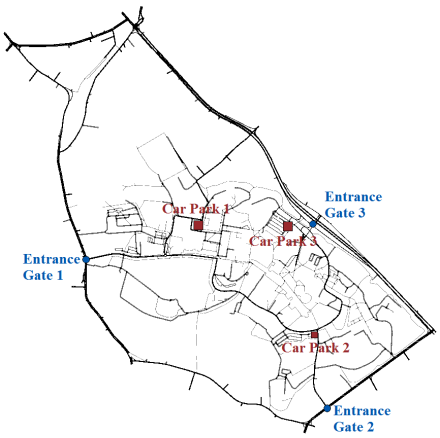

Fig. 4. UCD Belfield campus.

In regards to case (ii), the optimisation problem that we solved related to minimising CO_2 emissions. One of three possible cost functions, each function quantifying CO_2 emission levels associated with one (out of a potential three) categories of vehicle, was allocated to cars when they entered the campus road network for the first time. (In other words, a category of vehicle was allocated to cars when they entered the campus road network for the first time, each category of vehicle carrying with it an associated cost function.) These three categories of vehicles were [34, Table 1]: (a) petrol vehicles with engine capacity less than 1400 cc and European emission standard Euro 6 (code R007); (b) petrol vehicles with engine capacity between 1400 and 2000 cc and European emission standard Euro 6 (code R014); and (c) petrol vehicles with engine capacity greater than 2000 cc and European emission standard Euro 6 (code R021). At each time step, (4) was then solved, yielding a reference speed for Algorithm 2. An additional constraint applied to the problem was that the speed given by the solution would not exceed the maximum allowed speed on the campus roads. Specific descriptions of the three utilised cost functions were $h_i(v_i) = \frac{a+bv_i+cv_i^2+dv_i^3}{v_i}$, where the coefficients a, b, c and d are provided in Table I. These expressions yield grams of CO_2 emitted per kilometre. Further details regarding these cost functions can be found in [34, Sections 5.2, 5.4, 6.2] and [9, Section IV.A].

TABLE I
EMISSION FACTORS FOR DIFFERENT CATEGORIES OF VEHICLES

Category	a [g/h]	b [g/km]	c [g*h/km^2]	d [g*h^2/km^3]
R007	2260.6	31.583	0.29263	0.0030199
R014	2532.4	68.842	-0.43167	0.0066776
R021	3747.3	105.71	-0.8527	0.012264

A. Simulation Results

A simulation was performed to test convergence of the SASs. Fig's. 5 and 6 show the time evolution of individual vehicles' speeds, for test cases (i) and (ii), respectively. As expected, Fig's. 5 and 6 illustrate that the algorithms guide the speeds of the vehicles towards a common value. More precisely, as proven in the appendix, Algorithm 1 allows vehicles' speeds to converge towards an average value, while Algorithm 2 allows vehicles speed to converge towards v_{ref}. The dips in the plot lines in the figures are due to vehicles braking at roundabouts, traffic lights, etc. and the vehicles registering speeds of 0km/h are stopped at traffic lights, intersections, etc. The figures illustrate the results from the commencement of the simulation, to 400s from commencement.[5]

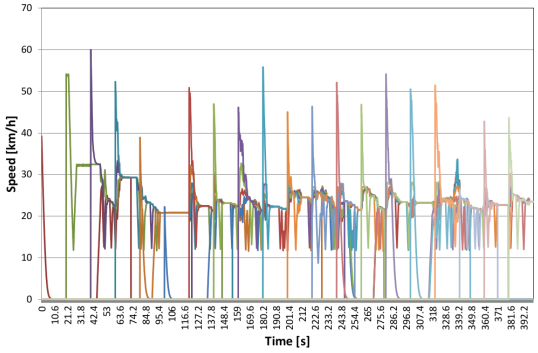

Fig. 5. Case (i): leaderless SAS. Individual vehicles' speeds versus time.

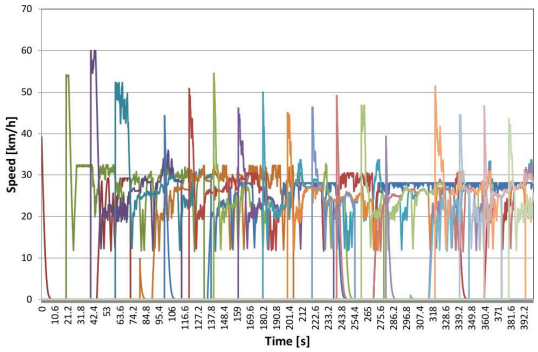

Fig. 6. Case (ii): SAS with leader. Individual vehicles' speeds versus time.

Table II compares total network CO_2 emissions over single simulation runs, as well as average CO_2 emissions per vehicle, for test cases (i) and (ii). Solving the optimisation problem of minimising CO_2 emissions to obtain reference signals for Algorithm 2 yielded a lower total network emissions quantity compared to test case (i) in which Algorithm 1 was applied. Note that, as per [34, Sections 5.2, 5.4, 6.2] and [9, Section IV.A], the cost functions used in case (ii) are only relevant when vehicles are travelling at a minimum of 5km/h. Thus, emission levels of individual cars over time step intervals when they weren't travelling at speeds above this threshold weren't included in the total network emission count. This applied for each test case so that comparisons between the two test cases could be made. Also recall that the speed limit on the campus is low and thus the speed regulation band in regards to the cost function curves is small. Finally, recall that, by the conclusion of each simulation run, sixty vehicles in total had entered the network, i.e. sixty vehicles for test case (i) and another sixty for test case (ii). The total network CO_2 emission values provided in Table II thus equate to the sum of

[5]The portions of results presented in Fig's. 5 and 6 are representative of the results obtained from the entire simulation runs.

emissions released from sixty individual cars over the course of their journeys, in each test case. The average CO_2 emissions per vehicle for each test case was calculated by dividing total network emissions by sixty.

TABLE II
CO_2 EMISSIONS

Test Case	Total Network (kg)	Per Vehicle (g)
(i) Leaderless SAS	31.95	532.5
(ii) SAS with Leader	28.12	468.67

B. Hardware-in-the-Loop

Eventual implementations of many intelligent transportation strategies, like the SASs reported in this work, are intended to be carried out on a large scale in the real world. For methodologies in their earlier stages of development, however, large fleets of real equipped test vehicles are expensive to come by, or are simply not practical. This dilemma led to the development of the hardware-in-the-loop (HIL) testing platform described in [30] (and seen also in https://www.youtube.com/watch?v=tCX2GLn1pnM). The objective of the HIL platform is to merge large-scale simulation and proof-of-concept vehicles by "embedding" real, equipped vehicles (being driven by real drivers) into SUMO. As such, emulations consisting of the real vehicles and potentially thousands of simulated cars are able to be run in realtime, and the drivers of the real vehicles are presented with an opportunity to experience first-hand what it feels like to travel in a large-scale, connected scenario and to try out the new intelligent transportation technology that is being developed.

An overview of the platform's architecture is given in Fig. 7. Information of interest from the real vehicle's onboard computer, such as the vehicle's speed, is obtained from the vehicle's OBD-II diagnostic connector using an adaptor and smartphone, and is communicated to a workstation computer over a cellular network. The workstation computer hosts a server for communication with the smartphone; and also hosts SUMO, and the application algorithm itself. Once the required computations are performed by the workstation computer using the combination of data from the simulated traffic and the real vehicle, then updates are sent to both the simulated and real traffic. In the case of a real vehicle, these updates are received by the smartphone travelling with the driver.

Thus, next, we utilised the HIL platform to test our SASs with a real driver. Using otherwise exactly the same setup on the UCD campus that was described above, we this time also "embedded" a real vehicle with driver into our emulations. Our real vehicle is pictured in Fig. 8 on the right. Our driver was requested to complete some circuits through the UCD campus, i.e. on topographically the same network that was imported into SUMO and that the simulated vehicles were also "driving" on. The test was conducted at a time during which there was almost no other real activity on the route around the campus (i.e. other real vehicles, pedestrians, etc.). The advised speeds issued by the workstation computer were provided to the driver visually on the smartphone carried in the real vehicle, and on

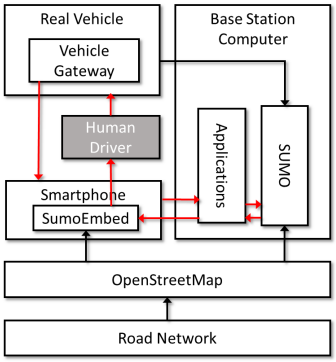

Fig. 7. Hardware-in-the-loop platform architecture [30].

Fig. 8. Toyota prius.

which we implemented the in-car algorithm. It was ideal to choose a time and day to perform our experiment when little real traffic was on the road that would otherwise complicate the setup.[6] The time step size used in our emulations remained set at 0.1s; however, information was only exchanged every 1s between the workstation computer and the smartphone carried in the real vehicle.

Fig's. 9 and 10 illustrate the advised speeds that the SASs issued the real driver (i.e. the top, red data line in each figure), which were obtained from implementations of Algorithms 1 and 2 on the campus network of traffic, now consisting of the simulated vehicles *and* the real car. Fig's. 9 and 10 also indicate the difference between the speeds at which the real driver was travelling minus the advised speeds (i.e. the bottom, green data lines). Fig. 9 relates to test case (i), while Fig. 10 corresponds to test case (ii). In our HIL implementation, we decided to not show any advised speed to the driver if the speed of the real car was below 20km/h. This choice was made to not distract the driver when he was driving at low speeds to, for example, decelerate due to pedestrians, or approach roundabouts or traffic lights. This absence of indication is the reason why there are some gaps in the data lines, indicating that no advice was given over certain time intervals. As can be seen from the bottom, green data lines, the driver followed the speed advice with less than a $\pm$2km/h error.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we presented two SASs that allow vehicles to achieve a common driving speed by implementing consensus

[6]The HIL platform is instrumented to enable the visual inspection of the full state of the simulation from the car via an additional tablet.

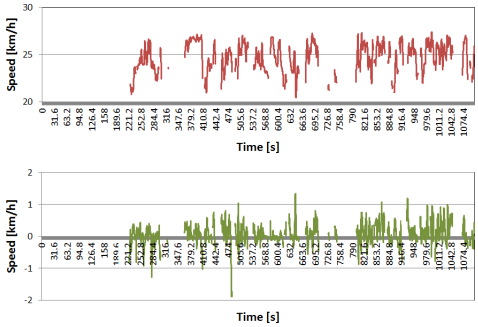

Fig. 9. Case (i): leaderless SAS. Real vehicle's advised speed (top line), and current minus advised speed (bottom line), filtered, versus time.

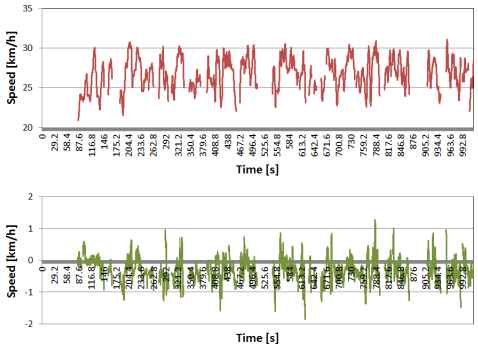

Fig. 10. Case (ii): SAS with leader. Real vehicle's advised speed (top line), and current minus advised speed (bottom line), filtered, versus time.

algorithms evolving on parallel network topologies of connected vehicles. The use of these parallel networks permitted the obfuscation of, via some noise, the signal received by each car. As a result, a single car in the network does not see the exact state of the other vehicles participating in the service. The algorithms presented here were also rigorously proven and, in particular, new results on the consensus of multi-layer networks modelled via stochastic differential equations were presented. Emulations of the SASs were performed via both pure simulation and via HIL techniques. Both validation methods showed that the algorithms effectively allowed vehicles to achieve a common driving speed. In regards to the HIL platform and experimentation with a real driver, the driver was able to follow the speed advice to within a range of $\pm$2km/h. An area of open research is to next study the effects of disobedient drivers or driver groups on the SASs presented. A further next step is to implement the SASs presented here more fully, to evaluate their performance in the real world with greater scope and to integrate this system with other services under development, such as the distributed congestion system recently introduced in [39].

Acknowledgment

The authors would like to thank Dr. Rodrigo H. Ordóñez-Hurtado for his assistance with the HIL experiments.

APPENDIX

In this appendix, we provide new results on the consensus of networks modelled by stochastic differential equations. Such results provide the theoretical foundations for the convergence of the SASs introduced in this paper. Specifically, we will first give a sufficient condition for the consensus of a stochastic network. Then, we consider the case where a network node receives, as input, an external reference signal and, in this case, we give a sufficient condition for all the network nodes to converge onto such signal.

NOTATION

We denote by I_n the $n \times n$ identity matrix and by $1_{n\times m}$ the $n \times m$ matrix having all of its elements equal to 1. The vector/matrix Frobenius norm will be denoted by $\|\cdot\|_F$ and the vector/matrix Euclidean norm will be denoted by $|\cdot|$. The Kronecker (or direct) product will be denoted by $\otimes$. The trace of a square matrix, say A, will be denoted by $tr\{A\}$ and its smallest (largest) eigenvalue will be denoted by $\lambda_{\min}(A)$ ($\lambda_{\max}(A)$). Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph with $\mathcal{V}$ being the set of nodes and $\mathcal{E}$ being the set of edges. Let N be the number of nodes in the network. Then (see e.g. [40]) the Laplacian matrix associated to the graph, $L := (l_{ij})_{i,j=1,\dots,N}$, is symmetric and we will denote with λ_i, $i = 1, \dots, N$, its eigenvalues.

MATHEMATICAL TOOLS

Consider an n-dimensional stochastic differential equation of the form

$$dx = f(t,x)dt + g(t,x)db, \tag{6}$$

where: (i) $x \in \mathbb{R}^n$ is the state variable; (ii) $f : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^n$ belongs to $\mathcal{C}^2$; (iii) $g : \mathbb{R}^+ \times \mathbb{R}^n \to \mathbb{R}^n$ belongs to $\mathcal{C}$; (iv) b is a 1-dimensional Brownian motion. Throughout this paper we will assume that, for any given initial condition, (6) has a unique global solution; see e.g. [25]. We will also assume that $f(t,0) = g(t,0) = 0$ and the solution $x = 0$ will be said to be the *trivial solution* of (6). Following [41], [42], we say that a sequence of stochastic variables $\{V_1, V_2, \dots\}$ converges almost surely (a.s.) to the stochastic variable V if $\mathbb{P}(\{w : \lim_{n\to+\infty} V_n(w) = V(w)\}) = 1$. That is, convergence happens with probability 1 ($\mathbb{P} = 1$). We are now ready to give the following definition; see [25].

Definition 1. *The trivial solution of (6) is said to be almost surely exponentially stable if, for all* $x \in \mathbb{R}^n$, $\lim_{t\to+\infty} \sup \frac{1}{t} \log(|x(t)|) < 0$, *a.s.*

Let $V(t,x) \in \mathcal{C}^{1\times 2}$ (i.e. $V(t,x)$ is twice differentiable in x and differentiable in t) and let: (i) $LV(t,x) = V_t(t,x) + V_x(t,x)f(t,x) + \frac{1}{2}tr\left\{g(t,x)^T V_{xx} g(t,x)\right\}$;(ii) $V_x = [V_{x_1}, \dots, V_{x_n}]$; (iii) V_{xx} be the $n \times n$ dimensional matrix having as element ij $V_{x_i x_j}$ (where $V_{x_i} := \partial V(t,x)/\partial x_i$ and $V_{x_i x_j} := \partial^2 V(t,x)/\partial x_j \partial x_i$). Then, the following result from [25] holds.

Theorem 1. *Assume that there exists a non-negative function* $V(t,x) \in \mathcal{C}^{1\times 2}$ *and constants* $p > 0$, $c_1 > 0$, $c_2 \in \mathbb{R}$, $c_3 \geq 0$ *such that* $\forall x \neq 0$ *and* $\forall t \in \mathbb{R}^+$: **(H1)** $c_1 |x|^p \leq V(t,x)^p$; **(H2)** $LV(t,x) \leq c_2 V(t,x)$; **(H3)** $|V_x(t,x)g(t,x)|^2 \geq c_3 V(t,x)^2$. *Then:* $\lim_{t\to+\infty} \sup \frac{1}{t} \log(|x(t)|) \leq -\frac{c_3 - 2c_2}{p}$, *a.s.*. *In particular, if* $c_3 > 2c_2$, *then the trivial solution of (6) is almost surely exponentially stable.*

STOCHASTIC NETWORKS AND CONSENSUS

We will first consider networks modeled by the following set of stochastic differential equations

$$dx_i = \left[\sigma \sum_{j\in\mathcal{N}_i}(x_j - x_i)\right] dt + \sigma^* \sum_{j\in\mathcal{N}_i^*}(x_j - x_i)\, db, \quad (7)$$

with initial conditions $x_i(0) = x_{i,0}$, $x_i \in \mathbb{R}$, $i = 1,\dots,N$, $\sigma > 0$ and $\sigma^* > 0$. Such an equation corresponds to the dynamics of a network consisting of two layers [24]. Note that the two layers might have different topologies and that one of the layers is affected by noise. In the rest of the paper, we will refer to the noise-free layer as *communication layer*, while the layer affected by noise will be termed as *noise-diffusion layer*. In the equation above, the set of neighbours of node i on the communication layer is denoted by $\mathcal{N}_i$, while the set of neighbours of the same node on the noise-diffusion layer is denoted by $\mathcal{N}_i^*$. Network (7) can be written in compact form as $dX = -\sigma LX dt - \sigma^* L^* X db$, where $X := [x_1,\dots,x_N]^T$, L is the Laplacian of the communication layer, and L^* is the Laplacian of the noise-diffusion layer. In what follows, we will denote by λ_N^* the largest eigenvalue of L^*, while λ_2^* will its algebraic connectivity. We will also consider the case where one network node of (7) is *pinned* (on the communication layer) to an exogenous constant signal, $x_r \in \mathbb{R}$. Without loss of generality, we can assume that the first network node is the one pinned to x_r. The resulting network model from (7) is then

$$\begin{aligned} dx_1 &= \left[\sigma\textstyle\sum_{j\in\mathcal{N}_1}(x_j - x_1) + \sigma\varepsilon(x_r - x_1)\right] dt + \\ &\quad + \left[\sigma^* \textstyle\sum_{j\in\mathcal{N}_1^*}(x_j - x_1)\right] db, \\ dx_i &= \left[\sigma\textstyle\sum_{j\in\mathcal{N}_i}(x_j - x_i)\right] dt + \sigma^* \textstyle\sum_{j\in\mathcal{N}_i^*}(x_j - x_i)\, db, \end{aligned} \quad (8)$$

$\forall i = 2,\dots,N$. We say that (7) achieves stochastic consensus if the following definition is fulfilled.

Definition 2. *Let $\bar{s} = \frac{1}{N}\sum_{i=1}^N x_{i,0}$. We will say that network (7) achieves stochastic consensus if $\lim_{t\to+\infty}\sup\frac{1}{t}\log(|x_i(t) - \bar{s}|) < 0$, a.s., $\forall i = 1,\dots,N$.*

That is, Definition 2 essentially means that all the network nodes (almost surely) agree onto the average of their initial conditions. With the following definition, instead, the agreement value will be the reference signal x_r.

Definition 3. *We will say that network (8) achieves complete stochastic consensus onto x_r if $\lim_{t\to+\infty}\sup\frac{1}{t}\log(|x_i(t) - x_r|) < 0$, a.s., $\forall i = 1,\dots,N$.*

Consensus of Network (7)

Theorem 2. *Assume that for network (7) the following condition is satisfied: $\sigma^{*2}\left(\lambda_2^{*2} - \frac{(\lambda_N^*)^2}{2}\right) > -\sigma\lambda_2$. Then, (7) achieves complete stochastic consensus.*

Proof. Let $S := 1_N \otimes \bar{s}$ and let $e = X - S$. Then, the error dynamics can be written as

$$de = \left[\tilde{F}(t,e)\right] dt + \left[\tilde{G}(t,e)\right] db, \quad (9)$$

where: (i) $\tilde{F}(t,e) = -\sigma L(e+S) = -\sigma Le$; (ii) $\tilde{G}(e) = -\sigma^* L^*(e+S) = -\sigma^* L^* e$. Note that $e = 0$ is the trivial solution for (9) and thus we can use Theorem 1 to prove our result. To this aim, let $V(t,e) = V(e) = \frac{1}{2}e^T e$. Then, we need to show that there exists $c_2 \in \mathbb{R}$, $c_3 \geq 0$, such that $c_3 > 2c_2$. We will now estimate $LV(e)$ and $\left|V_e(e)\tilde{G}(t,e)\right|^2$.

Estimate of $LV(e)$. In order to compute this term, first note that $V_t(e) = 0$. Let's now compute the term $V_e(e)\tilde{F}(t,e)$. We have $V_e(e)\tilde{F}(t,e) = -\sigma e^T Le$. Moreover, $V_e(e)\tilde{F}(t,e) \leq -\sigma min_{e\neq 0}\left\{e^T Le\right\}$, and since $e^T 1_N = 0$, then $\min_{e\neq 0}\left\{e^T Le\right\} = \lambda_2 e^T e$. That is,

$$V_e(e)\tilde{F}(t,e) \leq -\sigma\lambda_2 e^T e = 2(-\sigma\lambda_2)V(e). \quad (10)$$

The next step to estimate $LV(e)$ is that of computing $\frac{1}{2}tr\left\{\tilde{G}(e)^T V_{ee}\tilde{G}(e)\right\}$. Now, since $V_{ee}(e) = I_N$, such a term becomes $\frac{1}{2}tr\left\{\tilde{G}(t,e)^T\tilde{G}(t,e)\right\}$. Recall that for any matrix, say A, we have $\|A\|_F^2 = tr\left\{A^T A\right\}$. That is, $\left(tr\left\{(\sigma^*)^2 e^T (L^*)^T (L^*) e\right\}\right)^{1/2} = \sigma^*\left\|L^* e\right\|_F = \sigma^*|L^* e|$, where the last equality follows from the fact that, for any vector, say a, it holds that $\|a\|_F = |a|$. Therefore, we have $\frac{1}{2}\left(tr\left\{\tilde{G}(e)^T V_{ee}\tilde{G}(e)\right\}\right) = \frac{1}{2}(\sigma^*)^2|L^* e|^2 = \frac{1}{2}(\sigma^*)^2 e^T (L^*)^T (L^*) e$, from which we get

$$\frac{1}{2}tr\left\{\tilde{G}(t,e)^T V_{ee}\tilde{G}(t,e)\right\} \leq (\sigma^*)^2(\lambda_N^*)^2 V(e). \quad (11)$$

Combining (11) and (10), we finally have $LV(e) \leq \left(2(-\sigma\lambda_2) + (\sigma^*)^2(\lambda_N^*)^2\right)V(e)$.

Estimate of $\left|V_e(e)\tilde{G}(t,e)\right|^2$. This follows after noticing that $\left|V_e(e)\tilde{G}(t,e)\right| = \sigma^* e^T L^* e \geq \sigma^*\lambda_2^* e^T e$, and hence $\left|V_e(e)\tilde{G}(t,e)\right|^2 \geq (\sigma^*)^2(\lambda_2^*)^2(e^T e)^2 = 4(\sigma^*)^2(\lambda_2^*)^2 V(e)^2$. We can then conclude the proof by noticing that by hypotheses $4(\sigma^*)^2(\lambda_2^*)^2 > 2\left(2(-\sigma\lambda_2) + (\sigma^*)^2(\lambda_N^*)^2\right)$. Therefore, by means of Theorem 1, we have that $\lim_{t\to+\infty}\sup\frac{1}{t}\log(|e(t)|) < 0$, a.s., proving the result. □

Consensus of Network (8) onto x_r

Let $\tilde{L}$ be the $N \times N$ matrix obtained from a network Laplacian, L, as follows:

$$\tilde{L} := \begin{bmatrix} l_{11}+\varepsilon & l_{12} & \dots & l_{1N} \\ l_{21} & l_{22} & \dots & l_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ l_{N1} & l_{N2} & \dots & l_{NN} \end{bmatrix},$$

with eigenvalues $\tilde{\lambda}_1 \leq \tilde{\lambda}_2 \leq \dots \leq \tilde{\lambda}_N$.

Theorem 3. *Assume that for network (8) the following condition is satisfied: $\sigma^{*2}\left(\lambda_2^{*2} - \frac{(\lambda_N^*)^2}{2}\right) > -\sigma\tilde{\lambda}_1$. Then, (8) achieves complete stochastic synchronisation onto x_r.*

Proof. Let $e_i := x_i - x_r$. Then, we have $de_1 = \left[\sigma\sum_{j\in\mathcal{N}_1}(e_j - e_1) - \sigma\varepsilon e_1\right] dt + \left[\sigma^*\sum_{j\in\mathcal{N}_1^*}(e_j - e_1)\right] db$, $de_i = \left[\sigma\sum_{j\in\mathcal{N}_i}(e_j - e_i)\right] dt + \sigma^*\sum_{j\in\mathcal{N}_i^*}(e_j - e_i)\, db$, $\forall i = 2,\dots,N$. Let: (i) $X_r := 1_N \otimes x_r$; (ii) $e = X - X_r$. Then,

the error dynamics can be written again in compact form as (9), this time with $\tilde{F}(t,e) := -\sigma\tilde{L}e$ and $\tilde{G}(t,e) := -\sigma^* L^* e$. Note that $e = 0$ is again the trivial solution for the error dynamics and thus we can use Theorem 1. To this aim, let again $V(t,e) = V(e) = \frac{1}{2}e^T e$.

Estimate of $LV(e)$. In this case we have $V_e(e)\tilde{F}(t,e) \leq -\sigma\tilde{\lambda}_1 e^T e = -2(\sigma\tilde{\lambda}_1)V(e)$. Moreover, following similar steps used to prove Theorem 2 we get $\frac{1}{2}tr\left\{\tilde{G}(t,e)^T V_{ee}\tilde{G}(t,e)\right\} \leq (\sigma^*)^2(\lambda_N^*)^2 V(e)$. Therefore $LV(e) \leq \left(-2\left(\sigma\tilde{\lambda}_1\right) + (\sigma^*)^2(\lambda_N^*)^2\right)V(e) := c_2 V(e)$.

Estimate of $\left|V_e(e)\tilde{G}(t,e)\right|^2$. We have $\left|V_e(e)\tilde{G}(t,e)\right|^2 \geq 4(\sigma^*\lambda_2^*)^2 V(e)^2$.

The result is proved since, by hypotheses, $4(\sigma^*\lambda_2^*)^2 > 2\left(-2\sigma\tilde{\lambda}_1 + (\sigma^*)^2(\lambda_N^*)^2\right)$. □

Convergence of the SASs

We show that convergence of the SASs immediately follows from Theorem 2 and Theorem 3. To this aim, let v_i be the speed of the i-th vehicle receiving signals from the base station. Then, the dynamics of the speed is governed by

$$\dot{v}_i = u_i(t), \tag{12}$$

where $u_i(t)$ is computed following: (i) Equations (1), (2) and (3) for the leaderless SAS; (ii) Equations (5), (2) and (3) for the SAS with leader. The results proven in the appendix are valid for any number of nodes $N > 1$ and hence convergence of the SASs is guaranteed for any number of vehicles.

Convergence of the Leaderless SAS

Corollary 1. *Assume that network (12) is controlled by $u_i(t)$ computed following (1), (2) and (3). Then, the network achieves consensus.*

Proof. The proof immediately follows by noticing that the dynamics of network (12) can be recast as the following stochastic differential equation $dv = -Lvdt - \sigma^* L^* v db$, with $v := [v_1, \ldots, v_N]^T$, and where: (i) L is the Laplacian matrix generated by (1); (ii) L^* is the Laplacian matrix generated by (2); (iii) db is a standard Brownian motion and it can be thought of as the derivative of the white noise (of intensity σ^*) injected by the base station, i.e. $w(t)$. Specifically, note that: (i) the network generated by (1) is connected ($\lambda_2 > 0$); (ii) the network generated by (2) has $\lambda_2^* = \lambda_N^* = N$. Now, the proof follows immediately by noticing that $(\sigma^*)^2\left((\lambda_2^*)^2 - \frac{(\lambda_N^*)^2}{2}\right) = (\sigma^*)^2\frac{N^2}{2} > 0$. Therefore, since λ_2 is positive, then the condition of Theorem 2 is satisfied. The result is then proved. □

Convergence of the SAS with Leader

Corollary 2. *Assume that network (12) is controlled by $u_i(t)$ computed following (5), (2) and (3). Then, the network achieves consensus onto v_{ref}.*

Proof. In order to prove this result, notice that network dynamics can be written as $dv = -\tilde{L}vdt - \sigma^* L^* v db$, where this time $\tilde{L}$ is the Laplacian matrix generated by (5). Note that, in this special case, the $N \times N$ matrix $\tilde{L}$ is given by: $\tilde{L} = \begin{bmatrix} \varepsilon & 0 & \ldots & 0 \\ 0 & \ldots & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & \ldots & 0 \end{bmatrix}$. Hence, $0 = \tilde{\lambda}_1 = \ldots = \tilde{\lambda}_{N-1} < \tilde{\lambda}_N = \varepsilon$. Thus, $\tilde{\lambda}_1 = 0$ while $\lambda_2^* = \lambda_N^* = N$. Since $(\sigma^*)^2\left((\lambda_2^*)^2 - \frac{(\lambda_N^*)^2}{2}\right) = (\sigma^*)^2\frac{N^2}{2} > 0$, then the condition of Theorem 3 is satisfied. The result is then proved. □

Finally we remark that, from the theoretical viewpoint, as shown in Corollary 1 and Corollary 2, convergence of the SASs is independent of the intensity of the noise being injected (i.e. on σ^*). However, in simulations, we found that convergence of the SASs depended on the specific numerical method being used and on its time-step [43].

References

[1] W. Griggs, G. Russo and R. Shorten, "Consensus with state obfuscation: an application to speed advisory systems," in *Proc. 2016 IEEE 19th International Conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, 2016, pp. 2506-2511.

[2] S. Jamson, O. Carsten, K. Chorlton and M. Fowkes, "Intelligent speed adaptation literature review and scoping study," ISA-TfL D1, University of Leeds, MIRA Ltd. and Transport for London, 2006.

[3] European Transport Safety Council, "Reducing traffic injuries resulting from excess and inappropriate speed," ISBN 90-801936-3-1, Brussels, Belgium, 1994.

[4] Transport Accident Commission, "Speed Statistics," Victoria State Government, Australia. (Last accessed: 26th April 2017). [Online]. Available: http://www.tac.vic.gov.au/road-safety/statistics/summaries/speed-statistics

[5] S. Kundu, "Flexible vehicle speed control algorithms for eco-driving," in *Proc. 2015 IEEE 82nd Vehicular Technology Conference*, Boston, MA, USA, 2015.

[6] W. J. Schakel and B. van Arem, "Improving traffic flow efficiency by in-car advice on lane, speed, and headway," *IEEE Trans. Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1597-1606, 2014.

[7] R. H. Ordóñez-Hurtado, W. M. Griggs, K. Massow and R. N. Shorten, "Intelligent speed advising based on cooperative traffic scenario determination," pp. 77-92, in: H. Waschl, I. Kolmanovsky, M. Steinbuch and L. del Re (Eds.), *Lecture Notes in Control and Information Sciences: Optimization and Optimal Control in Automotive Systems*, vol. 455, Springer, 2014.

[8] G. De Nunzio, C. C. de Wit and P. Moulin, "Urban traffic eco-driving: speed advisory tracking," in *Proc. 2014 IEEE 53rd Conference on Decision and Control*, Los Angeles, CA, USA, 2014, pp. 1747-1752.

[9] M. Liu, R. H. Ordóñez-Hurtado, F. Wirth, Y. Gu, E. Crisostomi and R. Shorten, "A distributed and privacy-aware speed advisory system for optimizing conventional and electric vehicle networks," *IEEE Trans. Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1308-1318, 2016.

[10] E. Ozatay, S. Onori, J. Wollaeger, U. Ozguner, G. Rizzoni, D. Filev, J. Michelini and S. Di Cairano, "Cloud-based velocity profile optimization for everyday driving: a dynamic-programming-based solution," *IEEE Trans. Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2491-2505, 2014.

[11] R. Gallen, N. Hautière and S. Glaser, "Advisory speed for intelligent speed adaptation in adverse conditions," in *Proc. 2010 IEEE Intelligent Vehicles Symposium*, San Diego, CA, USA, 2010, pp. 107-114.

[12] McKinsey & Company, "What's driving the connected car," Article, September 2014. (Last accessed: 26th April 2017). [Online]. Available: http://www.mckinsey.com/industries/automotive-and-assembly/our-insights/whats-driving-the-connected-car

[13] N. A. Lynch, *Distributed Algorithms*. San Francisco, CA, USA: Morgan Kaufmann, 1996.

[14] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Trans. Automatic Control*, vol. 50, no. 5, pp. 655-661, 2005.

[15] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[16] G. Russo, M. di Bernardo, and E. D. Sontag, "A contraction approach to the hierarchical analysis and design of networked systems," *IEEE Trans. Automatic Control*, vol. 58, no.5, pp. 1328–1331, 2013.

[17] G. Russo, M. di Bernardo and E. D. Sontag, "Stability of networked systems: a multi-scale approach using contraction," in *Proc. 2010 IEEE 49th Conference on Decision and Control*, Atlanta, GA, USA, 2010, pp. 6559-6564.

[18] M. di Bernardo, D. Liuzza and G. Russo, "Contraction analysis for a class of nondifferentiable systems with applications to stability and network synchronization," *SIAM J. Control and Optimization*, vol. 52, no. 5, pp. 3203-3227, 2014.

[19] E. Adell, A. Várhelyi, M. Alonso and J. Plaza, "Developing human–machine interaction components for a driver assistance system for safe speed and safe distance," *IET Intelligent Transport Systems*, vol. 2, no. 1, pp. 1–14, 2008.

[20] R. Gallen, N. Hautière, A. Cord and S. Glaser, "Supporting drivers in keeping safe speed in adverse weather conditions by mitigating the risk level," *IEEE Trans. Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1558-1571, 2013.

[21] S. Kundu, A. Singh, S. Kundu, C. Qiao and Y. Hou, "Vehicle speed control algorithms for data delivery and eco-driving," in *Proc. 2014 International Conference on Connected Vehicles and Expo*, Vienna, Austria, 2014, pp. 270-271.

[22] E. Paikari, L. Kattan, S. Tahmasseby and B. H. Far, "Modeling and simulation of advisory speed and re-routing strategies in connected vehicles systems for crash risk and travel time reduction," in *Proc. 2013 IEEE 26th Canadian Conference on Electrical and Computer Engineering*, Regina, Canada, 2013.

[23] Y. Gu, M. Liu, E. Crisostomi and R. Shorten, "Optimised consensus for highway speed limits via intelligent speed advisory systems," in *Proc. 2014 International Conference on Connected Vehicles and Expo*, Vienna, Austria, 2014, pp. 1052-1053.

[24] D. Burbano and M. di Bernardo, "Multilayer proportional-integral consensus of heterogeneous multi-agent systems," in *Proc. 2015 IEEE 54th Conference on Decision and Control*, Osaka, Japan, 2015, pp. 4854-4859.

[25] X. Mao, *Stochastic Differential Equations and Applications*. Horwood Publishing, 1997.

[26] J. K. Hedrick, M. Tomizuka and P. Varaiya, "Control issues in automated highway systems," *IEEE Control Systems*, vol. 14, no. 6, pp. 21–32, 1994.

[27] D. Swaroop and J. K. Hedrick, "String stability of interconnected systems," *IEEE Trans. Automatic Control*, vol. 41, no. 3, pp. 349–357, 1996.

[28] S. Klinge and R. H. Middleton, "Time headway requirements for string stability of homogeneous linear unidirectionally connected systems," in *Proc. 2009 IEEE 48th Conference on Decision and Control, and the 28th Chinese Control Conference*, Shanghai, China, 2009, pp. 1992–1997.

[29] S. Knorn and R. H. Middleton, "Stability of two-dimensional linear systems with singularities on the stability boundary using LMIs," *IEEE Trans. Automatic Control*, vol. 58, no. 10, pp. 2579–2590, 2013.

[30] W. M. Griggs, R. H. Ordóñez-Hurtado, E. Crisostomi, F. Häusler, K. Massow and R. N. Shorten, "A large-scale SUMO-based emulation platform," *IEEE Trans. Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3050-3059, 2015.

[31] D. Krajzewicz, J. Erdmann, M. Behrisch and L. Bieker, "Recent development and applications of SUMO – Simulation of Urban MObility," *International J. Advances in Systems and Measurements*, vol. 5, no. 3-4, pp. 128-138, 2012.

[32] R. Cogill, O. Gallay, W. Griggs, C. Lee, Z. Nabi, R. Ordonez, M. Rufli, R. Shorten, T. Tchrakian, R. Verago, F. Wirth and S. Zhuk, "Parked cars as a service delivery platform," in *Proc. 2014 International Conference on Connected Vehicles and Expo*, Vienna, Austria, 2014, pp. 138-143.

[33] CVXPY — What is CVXPY? (Last accessed: 9th January 2017). [Online]. Available: http://www.cvxpy.org/en/latest/tutorial/intro/index.html

[34] P. G. Boulter, T. J. Barlow and I. S. McCrae, "Emission factors 2009: report 3 – exhaust emission factors for road vehicles in the United Kingdom," PPR356, TRL Limited, 2009.

[35] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer and J.-P. Hubaux, "TraCI: an interface for coupling road traffic and network simulators," in *Proc. 11th Communications and Networking Simulation Symposium*, Ottawa, Canada, 2008, pp. 155-163.

[36] JOSM (Java OpenStreetMap Editor). (Last accessed: 10 September 2013). [Online]. Available at: http://josm.openstreetmap.de.

[37] XMLStarlet. (Last accessed: 10 September 2013). [Online]. Available at: http://xmlstar.sourceforge.net.

[38] SUMO (Simulation of Urban MObility). (Last accessed: 2nd March 2016). [Online]. Available: www.dlr.de/ts/sumo/en/

[39] S. Sinnott, R. Ordóñez-Hurtado, G. Russo and R. Shorten, "On the design of a route parsing engine for connected vehicles with applications to congestion management systems," in *Proc. 2016 IEEE 19th International Conference on Intelligent Transportation Systems*, Rio de Janeiro, Brazil, 2016, pp. 1586-1591.

[40] C. Godsil and G. Royle, *Algebraic Graph Theory*. New York, NY, USA: Springer-Verlag, 2001.

[41] A. F. Karr, *Probability*. New York, NY, USA: Springer-Verlag, 1993.

[42] V. K. Rohatgi, *An Introduction to Probability Theory and Mathematical Statistics*. John Wiley & Sons, 1976.

[43] D. J. Higham, "An algorithmic introduction to numerical simulation of stochastic differential equations," *SIAM Review*, vol. 43, no. 3, pp. 525-546, 2001.

Wynita Griggs received a Ph.D. degree in engineering from the Australian National University, Canberra, Australia in 2007. Between 2008 and 2015, she was a Postdoctoral Research Fellow at the Hamilton Institute, National University of Ireland Maynooth, Ireland. She is currently a Research Scientist at University College Dublin, Dublin, Ireland. Her research interests include stability theory with applications to feedback control systems and smart transportation.

Giovanni Russo obtained his Ph.D. from the University of Naples Federico II in 2010. During this time, his work focused on the stability of nonlinear dynamical systems with applications to networked control. Motivated by a desire to apply his results to real world systems, soon after his Ph.D., Giovanni started to work on automatic transportation systems. From 2012 to 2015, he was the lead system engineer and integrator of the Honolulu Rail Transit Project (HRTP). In 2015, Giovanni joined IBM Research Ireland. He is currently a member of the Board of Editors for the IEEE Transactions on Circuits and Systems I for topics related to Control Theory, Networks and Networked Systems.

Robert Shorten received a Ph.D. degree from University College Dublin (UCD), Dublin, Ireland, in 1996. From 1993 to 1996, he was the holder of a Marie Curie Fellowship at Daimler-Benz Research, Berlin, Germany to conduct research in the area of smart gearbox systems. Following a brief spell at the Centre for Systems Science, Yale University, working with Prof. K. S. Narendra, he returned to Ireland as the holder of a European Presidency Fellowship in 1997. He is a co-founder of the Hamilton Institute, National University of Ireland Maynooth, Ireland, where he was a full Professor until March 2013. From 2013-2015 he was a Senior Research Manager at IBM Research Ireland, Dublin, leading the Control and Optimisation activities at IBM Research in Dublin. He currently holds a dual appointment with IBM Research and University College Dublin, where he is Professor of Control Engineering and Decision Science.