Delft University of Technology

Delft Center for Systems and Control

Technical report 17-010

Multi-agent dynamic routing of a fleet of cybercars*

R. Luo, T.J.J. van den Boom, and B. De Schutter

If you want to cite this report, please use the following reference instead:
R. Luo, T.J.J. van den Boom, and B. De Schutter, "Multi-agent dynamic routing of a fleet of cybercars," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1340–1352, May 2018.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/17_010.html`

Multi-Agent Dynamic Routing of a Fleet of Cybercars

Renshi Luo, Ton J.J. van den Boom, and Bart De Schutter, *Senior Member, IEEE*

***Abstract*—Due to the lack of efficient control methods for a fleet of vehicles throughout a road network, the large-scale application of cybercars, which are fully automatic road vehicles providing on-demand and door-to-door transportation service, is still hindered. Although the fleet control problem for cybercars can be straightforwardly addressed in a centralized control setting, for reasons of scalability and fast computation, a centralized control method will not be tractable for the large-scale use of cybercars in the future. In this paper, we focus on the dynamic routing of a fleet of cybercars considering minimization of the combined system cost including the total time spent and the total energy consumption by all cybercars. We first propose a model of the dynamics and the energy consumption of a fleet of cybercars based on a description of the dynamics of every single cybercar and the states of the road network. After that, we propose several tractable and scalable multi-agent control methods including multi-agent model predictive control and parameterized control for the dynamic routing of cybercars. Finally, experiments by means of numerical simulations illustrate the performance of the proposed control methods.**

***Index Terms*—Cybercar, intelligent transportation system, multi-agent control, model predictive control, parameterized control**

I. Introduction

TO deal with the problems caused by the increasing use of private cars, such as frequent congestion, increasing energy consumption and pollution, etc, a new intelligent transportation system called cybernetic transportation system, which is exclusively formed by a fleet of fully automatic electric vehicles (i.e., cybercars), has been proposed [1]. Due to the high flexibility and reactivity of cybercars, a cybernetic transportation system is able to offer better personal mobility than conventional public transportation systems [2]. Besides, a cybernetic transportation system is more competitive in terms of energy consumption than public transportation systems on a per passenger-km basis [3]. In fact, there have been many projects, such as the European project CyberCars [4], CyberCars-2, CyberC3 [5] and CyberMove, dedicated to developing such a cybernetic transportation system, and there have been many operational cybernetic transportation systems, such as GRT (i.e., group rapid transit) at Schiphol Airport in Amsterdam, PRT (i.e., personal rapid transit) in Masdar City in the United Arab Emirates, and Ultra (i.e., urban light transit) at Heathrow Airport in London.

So far, there have been many automated driving technologies available for individual vehicles [6], such as automated lane change [7] and adaptive cruise control [8]. However, there is still no efficient method for the control of a fleet of vehicles. Hence, the large-scale application of cybercars is still hindered.

Actually, the fleet control problem for cybercars has been considered in the literature. More specifically, the problem was studied in [3] from a conceptual point of view and a centralized fleet management system for cybercars was proposed. However, that paper only focused on the design of the system architecture without addressing the fleet control problem explicitly. In [9], a novel open-control concept that merges both centralized and decentralized control approaches for cybercars was proposed. However, that paper only focused on demonstrating how a cybernetic transportation system may benefit from the open-control concept in dealing with perturbations caused by the environment, while it did not introduce a specific algorithm for fleet control. In [10], a specific instance of the fleet control of cybercars, i.e.,the vehicle routing problem for an on-demand transportation system, was studied. However, that paper focused on solving the vehicle routing problem on a daily basis without considering the real-time conditions of the network. In contrast, in our research, we explore the dynamic routing problem of cybercars by considering the dynamics and the energy consumption of every cybercar according to the real-time conditions of the road network. We develop efficient strategies for the dynamic routing of cybercars so that the total costs for all cybercars, including the total time spent (TTS) and the total energy consumption (TEC), are minimized.

By directly incorporating system constraints as inequalities in the control problem formulation, model predictive control (MPC) has shown to be promising for control of road traffic networks [11]–[13]. However, for reasons of scalability and fast computation, centralized MPC will not be tractable for the control of large-scale cybernetic transportation systems. Therefore, multi-agent control methods have to be employed.

In multi-agent MPC, the overall control problem is first divided into a set of subproblems, which are assigned to different agents. The agents then determine their control actions by solving their local subproblems and coordinating with others [14]. Multi-agent MPC algorithms have been applied to power generation systems [15], chemical processes [16], and supply chains [17]. In this paper, the parallel multi-agent MPC scheme presented in [18] is adapted and applied to the dynamic routing of cybercars.

Besides, in parameterized control, the control laws are parameterized and then the parameters are optimized with respect to the overall performance of the system. Parameterized control methods have been applied to control of freeway traffic [19], of robotic systems [20], and of baggage handling systems [21]. In this paper, several computationally fast and scalable

Manuscript received Month Day, Year; revised Month Day, Year. This work was supported by the China Scholarship Council under Grant 201207090001.

The authors are with Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands (email: r.luo@tudelft.nl; a.j.j.vandenboom@tudelft.nl; b.deschutter@tudelft.nl).

multi-agent parameterized control methods are proposed for the dynamic routing of cybercars.

Actually, in our previous work [22], we have proposed a model of the dynamics and the energy consumption of cybercars. However, in [22], we assumed that at any time the speed of each cybercar is determined by the equilibrium speed of all cybercars in the segment. In contrast, in this paper, we propose to use the dynamic speed equation, which is more accurate in capturing the speed change of a single cybercar in reality. Besides, the impact of the slope of each segment in the dynamics and energy consumption of cybercars was missing in [22] but is considered in the model provided in this paper. This makes the model provided in this paper more general.

With respect to the literature, the main contributions of this paper are addressing a specific instance of the fleet control problem of cybercars, i.e., the dynamic routing of a fleet of cybercars, providing a more general and more accurate modeling of the dynamics and the energy consumption of cybercars, and proposing several tractable and scalable multi-agent control methods to solve the problem.

This paper is organized as follows. In Section II, we describe the general dynamic routing problem of cybercars. In Section III, the model of the dynamics and the energy consumption of cybercars is presented. In Section IV, the dynamic routing problem of cybercars is formulated. In Section V and Section VI, we propose a multi-agent MPC scheme and six different parameterized control methods for the dynamic routing of cybercars, respectively. Section VII presents a simulation study where the performance of all proposed methods is assessed and compared. Finally, in Section VIII, the main contributions of this paper are summarized, and some ideas of future work are presented.

II. PROBLEM DESCRIPTION

We consider a cybernetic transportation network (i.e. only open to cybercars) consisting of a set of roads and a set of intersections. For the sake of simplicity, we refer to an intersection as a 'node', and a road section between two intersections as a 'link'. Each link is divided into a number of segments that have typical lengths ranging between 50 m and 100 m. We assume that, at any time, the equilibrium speed[1] of all cybercars in a segment is determined by the traffic density in that segment, while the actual speed of each cybercar is determined by its previous speed and its current equilibrium speed. Note that the equilibrium speed-flow relation (i.e., fundamental diagram of traffic flow) exists on a road section in urban environments [24], and that the average flow dynamics of cybercars can be modeled in a similar way to that used for modeling the average flow dynamics of human-driven vehicles but with different values for the model parameters. Moreover, we assume each segment has a maximum capacity[2] of cybercars at the same time. More specifically, if the number of cybercars in a segment reaches or exceeds the maximum capacity, that segment will be blocked. A blocked segment will be unblocked immediately when the number of cybercars in that segment becomes lower than the maximum capacity. At any time, the energy consumption of a cybercar is a function of its velocity, its acceleration (or deceleration), as well as its position (related to its potential-energy change in case of link with slope). Without loss of generality, we assume the origins and the destinations of all cybercars are always at nodes. Besides, we also assume that no cybercar can cover a distance longer than the length of a segment within one simulation time interval. Finally, we assume there are higher-level controllers assigning transport service requests (including starting time, origin and destination) to each cybercar and we only focus on solving the dynamic routing problem for all cybercars with the transport service requests given.

[1] In macroscopic traffic flow models where the equilibrium speed-flow relationship of traffic flows is described, the equilibrium speed is an aggregated traffic variable representing the average speed of vehicles over a time period or over an area [23].

[2] We assume that when the maximum capacity of a segment is reached, the traffic density in the segment is still less than the jam density of the segment. Therefore, the equilibrium speed of cybercars in a segment will never be 0 m/s.

TABLE I
NOMENCLATURE OF THE MATHEMATICAL SYMBOLS

Symbol	Definition
T	length of the simulation time interval
k	discrete-time step counter
$T_{\text{start},i}$	starting time of cybercar i
$T_{\text{stop},i}$	arrive time of cybercar i at its destination
$v_i(k)$	speed (measured along the longitudinal direction of a link) of cybercar i at time kT
$l_i(k)$	link in which cybercar i is running at time kT
$s_i(k)$	segment in which cybercar i is running at time kT
$x_i(k)$	position (measured along the longitudinal direction of a link) of cybercar i in $l_i(k)$ at time kT
$l_{\text{final},i}$	final link of cybercar i (the end of $l_{\text{final},i}$ is the destination of cybercar i)
$r_i(k)$	selected route of cybercar i at time kT
$u_i(k)$	next link of cybercar i at time kT
$p_{m,j}^{\text{start}}$	position of the starting point of segment m of link j
$p_{m,j}^{\text{end}}$	position of the end point of segment m of link j
$v_{\text{free},m,j}$	free-flow speed of segment m of link j
$N_{m,j}(k)$	number of cybercars in segment m of link j at time kT
$L_{m,j}$	length of segment m of link j
$\rho_{m,j}(k)$	traffic density of segment m of link j at time kT
$C_{m,j}$	maximum capacity of segment m of link j
$b_{m,j}(k)$	binary blocking signal of segment m of link j at time kT with $b_{m,j}(k)=1$ indicating that the segment is blocked
$\Delta_{m,j}(k)$	change of the number of cybercars in segment m of link l during one simulation interval at time kT

III. DISCRETE-TIME MODELING

A. Definitions

The definitions of the most important mathematical symbols used in this paper are presented in Table I, where $T_{\text{stop},i}$ for all i are initialized with sufficiently large numbers, and $\Delta_{m,j}(k)$ for all m and j are set to be 0 at the start of every simulation time interval.

B. Network Set-Up

In case of blocked departure links, cybercars are not able to enter the network even when the times at which they are due to depart have come. In order to model the queues of cybercars waiting at the origins due to blocked departure links, to each departure point, we introduce a virtual link with zero length

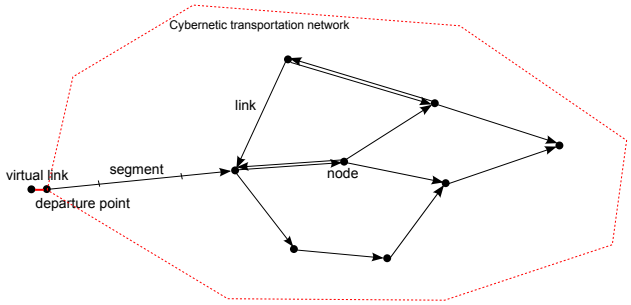

Fig. 1. Cybernetic transportation network

and infinite capacity. Without loss of generality, the layout of a cybernetic transportation network can then conceptually be represented by the graph shown in Figure 1, where a node is represented by a small solid circle while a link is represented by a directed line with the arrow indicating the heading direction.

C. Equilibrium speed-flow relationship considering slope

Vehicles tend to accelerate (decelerate) when going downhill (uphill). Based on the experimental results of geometric effects on the speeds of vehicles presented in [25], the impact of the slope on the free-flow speed of a segment can be modeled as follows:

$$v_{\text{free},m,j} = \begin{cases} v_{\text{free},m,j,0}\Big(1-\delta_{\text{up},m,j}\tan(\vartheta_{m,j})\Big), & \text{if } \vartheta_{m,j}\geq 0 \\ v_{\text{free},m,j,0}\Big(1+\delta_{\text{down},m,j}\tan(\vartheta_{m,j})\Big), & \text{if } \vartheta_{m,j}<0 \end{cases}$$

where $\vartheta_{m,j}$ denotes the angle of inclination of segment m of link j, $v_{\text{free},m,j,0}$ denotes the free-flow speed of vehicles in segment m of link j if that segment is flat, $\delta_{\text{up},m,j}$ and $\delta_{\text{down},m,j}$ are relative terms that denote the impact of each 1% downhill and uphill grade on the free-flow speed of cybercars in segment m of link j, respectively.

Besides, the critical traffic density of segment m of link j at which the maximal flow is obtained may also be influenced by the slope. Also inspired by [25], one possible way to model this influence is given by

$$\rho_{\text{crit},m,j} = \rho_{\text{crit},m,j,0}\Big(1+\alpha_{m,j}\tan(\vartheta_{m,j})\Big)$$

where

$$\rho_{\text{crit},m,j,0} = \frac{1\ [\text{veh}]}{h_{\text{con},m,j}v_{\text{free},m,j,0}+L_{\text{veh}}}$$

is the critical traffic density of segment m of link j if that segment would be flat [26], $h_{\text{con},m,j}$ is the constant time headway of automated vehicles on segment m of link j, L_{veh} is the average length of vehicles, and $\alpha_{m,j}$ is a model parameter. Note that $\alpha_{m,j}$ is a signed variable with a positive or negative sign depending on whether the slope is uphill or downhill. The value of $\alpha_{m,j}$ can be determined using identification techniques [27].

Finally, according to the macroscopic characteristics of semi-automated traffic presented in [26], the equilibrium speed-flow relationship of cybercars in a segment with slope is given by

$$V_{m,j}\big(\rho_{m,j}(k)\big) = \begin{cases} v_{\text{free},m,j}, & \text{if } \rho_{m,j}(k)\leq\rho_{\text{crit},m,j} \\ \frac{c_{m,j}}{\rho_{m,j}(k)}+d_{m,j}, & \text{if } \rho_{m,j}(k)>\rho_{\text{crit},m,j} \end{cases}$$

with

$$c_{m,j} = \frac{v_{\text{free},m,j}\cdot\rho_{\text{crit},m,j}\cdot\rho_{\text{jam},m,j}}{\rho_{\text{jam},m,j}-\rho_{\text{crit},m,j}}, \quad d_{m,j} = -\frac{v_{\text{free},m,j}\cdot\rho_{\text{crit},m,j}}{\rho_{\text{jam},m,j}-\rho_{\text{crit},m,j}}$$

where $\rho_{\text{jam},m,j}$ is the jam traffic density of segment m of link j, i.e. the density at which the traffic flow is 0 veh/h.

D. Speed Change of a Single Cybercar

In the discrete-time modeling framework, the speed of a cybercar is assumed to be fixed within one simulation time interval and is updated only at the end of the simulation time interval. In this paper, by assuming cybercar i is in segment m of link j at simulation time step k, the update of the speed of cybercar i is given by

$$v_i(k+1) = v_i(k)+\xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big)-v_i(k)\Big) \tag{1}$$

where ξ_i indicates how fast can cybercar i change its speed based on the difference of its desired speed and its current speed.. To be more specific, ξ_i is given by

$$\xi_i = \frac{a_{\max,i}T}{v_{\max,i}} \tag{2}$$

where $v_{\max,i}$ and $a_{\max,i}$ are the maximal speed and maximal acceleration rate of cybercar i, respectively. Note that the ordering of cybercars in a segment according to their positions could have been taken into account in the update of the speed of each cybercar, but that would cost many extra computations and there would not be much gain in modeling accuracy since the typical length of a segment (between 50 m and 100 m) is not that long.

E. Dynamics of a Single Cybercar

Each cybercar i enters the network at $T_{\text{start},i}$. After that, at each simulation time step kT, with $x_i(k)$, $v_i(k)$, $l_i(k)$, $s_i(k)$, $r_i(k)$ and $N_{m,j}(k)$, $\rho_{m,j}(k)$, $b_{m,j}(k)$ for all j and m given, the variables $x_i(k+1)$, $v_i(k+1)$, $l_i(k+1)$, and $s_i(k+1)$ of cybercar i need to be determined. As cybercar i may go from one segment (or link) to a different segment (or link) during one simulation time interval, the change of the number of vehicles in the segments due to the change of the position of cybercar i also needs to be captured.

From simulation time step kT to step $(k+1)T$, the update of the dynamics of a single cybercar i can be divided into five cases, which are characterized as follows:

- "**same segment, same link**": cybercar i stays in the same segment and the same link.
- "**different segments, same link**": cybercar i goes from one segment to the next one in the same link.
- "**desired link blocked**": cybercar i reaches the end of its current link, but its desired next link is blocked.
- "**different links**": cybercar i goes from its current link to its desired next link.

- "**arrival**": cybercar i arrives at its destination.

For the sake of simplicity of notation, in the following, we assume $l_i(k) = j$ and $s_i(k) = m$ when we describe the update of the dynamics of cybercar i in each of the cases.

First, the conditions for the case of **same segment, same link** are:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \Big[v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)\Big]T \le p^{\text{end}}_{m,j}$$

where the function $V_{m,j}(\cdot)$ describes how the equilibrium speed of cybercars in segment m of link j depends on the traffic density in that segment. One possible way to define $V_{m,j}(\cdot)$ has been given in the Section III-C. The dynamics of cybercar i are then updated by

$$v_i(k+1) \leftarrow v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)$$
$$x_i(k+1) \leftarrow x_i(k) + v_i(k+1)T$$
$$l_i(k+1) \leftarrow l_i(k)$$
$$s_i(k+1) \leftarrow s_i(k)$$

For the case of **different segments, same link**, the following conditions must be satisfied:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \Big[v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)\Big]T > p^{\text{end}}_{m,j}$$
$$b_{m+1,j}(k) = 0$$

In this case, cybercar i first runs at the speed $v_{\text{aux},i}(k) = v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)$ in the current segment. After reaching the end of the current segment, it enters the next segment and runs at $v_{\text{aux},i}(k) + \xi_i\Big(V_{m+1,j}\big(\rho_{m+1,j}(k)\big) - v_{\text{aux},i}(k)\Big)$ for the rest of the time interval. Then the dynamics of cybercar i are updated by

$$v_i(k+1) \leftarrow v_{\text{aux},i}(k) + \xi_i\Big(V_{m+1,j}\big(\rho_{m+1,j}(k)\big) - v_{\text{aux},i}(k)\Big)$$
$$x_i(k+1) \leftarrow p^{\text{end}}_{m,j} + v_i(k+1)\tau_i$$
$$l_i(k+1) \leftarrow l_i(k)$$
$$s_i(k+1) \leftarrow s_i(k) + 1$$

where τ_i denotes the remaining time during $[kT, (k+1)T]$ after cybercar i arrives at the end of the current segment m:

$$\tau_i = T - \frac{p^{\text{end}}_{m,j} - x_i(k)}{v_{\text{aux},i}(k)}$$

Besides, the changes of the number of cybercars in segment m and segment $m+1$ are captured by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{m+1,j} \leftarrow \Delta_{m+1,j} + 1$$

Next, the conditions for the case **desired link blocked** are given by

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \Big[v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)\Big]T > p^{\text{end}}_{m,j}$$
$$b_{1,j^*}(k) = 1$$

where j^* denotes the desired next link of cybercar i at kT. Given that the desired next link is currently blocked, cybercar i has to wait after arriving at the end of the current link. Therefore, the update of dynamics of cybercar i is given by

$$v_i(k+1) \leftarrow 0$$
$$x_i(k+1) \leftarrow p^{\text{end}}_{m,j}$$
$$l_i(k+1) \leftarrow l_i(k)$$
$$s_i(k+1) \leftarrow s_i(k)$$

The conditions for the case of **different links** are:

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \Big[v_i(k) + \xi_i\Big(V_{m,j}\big(\rho_{m,j}(k)\big) - v_i(k)\Big)\Big]T > p^{\text{end}}_{m,j}$$
$$b_{1,j^*}(k) = 0$$

In this case, cybercar i enters link j^*, and its dynamics are updated by

$$v_i(k+1) \leftarrow v_{\text{aux},i}(k) + \xi_i\Big(V_{1,j^*}\big(\rho_{1,j^*}(k)\big) - v_{\text{aux},i}(k)\Big)$$
$$x_i(k+1) \leftarrow v_i(k+1)\tau$$
$$l_i(k+1) \leftarrow u_i(k)$$
$$s_i(k+1) \leftarrow 1$$

The changes of the number of cybercars in the last segment of link j and the first segment of link j^* are then captured by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$
$$\Delta_{1,j^*} \leftarrow \Delta_{1,j^*} + 1$$

Finally, for the case of **arrival**, the conditions are given by

$$T_{\text{start},i} < (k+1)T$$
$$x_i(k) + \Big[v_i(k) + \xi_i\Big(V\big(\rho_{m,j}(k)\big) - v_i(k)\Big)\Big]T \ge p^{\text{end}}_{m,j}$$
$$l_{\text{final},i} = j$$

In this case, cybercar i reaches its destination and its arrival time $T_{\text{stop},i}$ is obtained by

$$T_{\text{stop},i} = kT + \frac{p^{\text{end}}_{m,j} - x_i(k)}{v_{\text{aux},i}(k)} \tag{3}$$

We assume cybercar i leaves the network after arriving at its destination. Then $\Delta_{m,j}$ is updated by

$$\Delta_{m,j} \leftarrow \Delta_{m,j} - 1$$

F. Dynamics of the Network

At every simulation time step, after the dynamics of all cybercars are updated, the states of the whole network are updated by

$$N_{m,j}(k+1) = N_{m,j}(k) + \Delta_{m,j}$$
$$\rho_{m,j}(k+1) = \frac{N_{m,j}(k+1)}{L_{m,j}}$$
$$b_{m,j}(k+1) = \mathbb{1}\Big(N_{m,j}(k+1) \ge C_{m,j}\Big)$$

where $\mathbb{1}(\cdot)$ is an indicator function defined by

$$\mathbb{1}(a) = \begin{cases} 1, & \text{if } a \text{ is true} \\ 0, & \text{else} \end{cases}$$

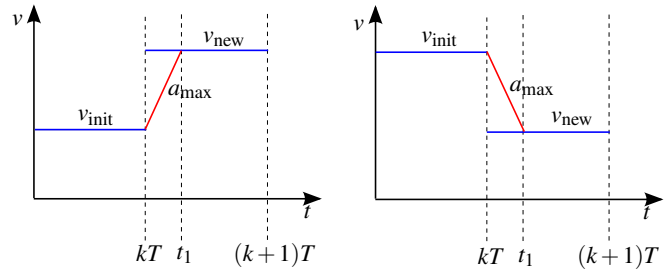

Fig. 2. Two possible cases of the speed change of a cybercar

G. Energy Consumption of a Single Cybercar

Generally, the energy consumption of a cybercar consists of five consuming factors [28], i.e., speeding up, air drag, rolling resistance, going uphill, and energy losses in the energy-conversion chain. Assuming each cybercar has the same acceleration and deceleration rate, we use the graphs in Figure 2 to calculate the first three consuming factors of a cybercar. For notational convenience, in this section, we drop the subscript indicating the index of the cybercar for all the variables. Note that in Figure 2, v_{init} and v_{new} respectively denote the speed of the cybercar at the beginning and the speed at the end of a simulation interval, and t_1 denotes the absolute time instant at which v_{new} is reached. According to (1) and (2), $t_1 < (k+1)T$ always holds. It should also be noted that the effects of acceleration and deceleration of a cybercar are non-negligible in the calculation of its energy consumption. Therefore, different from Section III-E where the dynamics of a cybercar are updated assuming a constant speed in every simulation interval, in this subsection, the acceleration and deceleration processes of a cybercar are approximated and then taken into account in the energy consumption calculation. In fact, acceleration and deceleration could also have been taken into account in the update of the dynamics of a cybercar in Section III-E, but the relative effect of that would be much smaller than for the energy consumption calculation.

According to Figure 2, the cybercar is first accelerating or decelerating at a rate $a_{\max}$ during $[kT, t_1]$, which we define as the acceleration-deceleration period. For the cybercar in this period, let $E_{\text{var\_kin}}$ denote the change of kinetic energy, $E_{\text{var\_air}}$ the energy consumption needed to overcome the air drag, and $E_{\text{var\_rol}}$ the energy consumption needed to overcome the rolling resistance. Based on [28], we have

$$E_{\text{var\_kin}} = M\int_{v_{\text{init}}}^{v_{\text{new}}} v\,dv = \frac{1}{2}M(v_{\text{new}}^2 - v_{\text{init}}^2) \tag{4}$$

$$E_{\text{var\_air}} = \int_{kT}^{t_1} \frac{1}{2}\rho_{\text{air}}A_{\text{front}}v^3\,dt = \frac{\rho_{\text{air}}A_{\text{front}}\left|v_{\text{new}}^4 - v_{\text{init}}^4\right|}{8a_{\max}} \tag{5}$$

$$E_{\text{var\_rol}} = \int_{kT}^{t_1} c_{\text{r}}Mgv\,dt = \frac{c_{\text{r}}Mg\left|v_{\text{new}}^2 - v_{\text{init}}^2\right|}{2a_{\max}} \tag{6}$$

where M and A_{front} denote respectively the mass and the effective frontal area of the cybercar, ρ_{air} denotes the air density, c_{r} denotes the rolling resistance coefficient, and g denotes the gravitational acceleration.

After the acceleration-deceleration period, the cybercar keeps a constant speed v_{new} for the rest of the simulation interval $[t_1, (k+1)T]$, which we define as the constant-speed period. In this period, the kinetic energy of the cybercar does not change. Then by defining $E_{\text{ct\_air}}$ and $E_{\text{ct\_rol}}$ respectively as the energy consumption of the cybercar to overcome the air drag and the rolling resistance during the constant-speed period, we have

$$E_{\text{ct\_air}} = \frac{1}{2}\rho_{\text{air}}A_{\text{front}}v_{\text{new}}^3 \cdot \left(T - \frac{|v_{\text{new}} - v_{\text{init}}|}{a_{\max}}\right) \tag{7}$$

$$E_{\text{ct\_rol}} = c_{\text{r}}Mgv_{\text{new}} \cdot \left(T - \frac{|v_{\text{new}} - v_{\text{init}}|}{a_{\max}}\right) \tag{8}$$

Furthermore, we consider the change of the potential energy of the cybercar, which is denoted by E_{pot}, during one simulation interval. If the cybercar stays in the same segment m of the same link j within the simulation interval, given the angle of inclination $\vartheta_{m,j}$ of the segment, we have

$$E_{\text{pot}} = Mg\sin(\vartheta_{m,j}) \cdot (x_{\text{new}} - x_{\text{init}}) \tag{9}$$

where x_{new} and x_{init} are the positions of the cybercar in the link at the beginning and the end of the simulation interval, respectively. If the cybercar goes from one segment to another within the simulation interval, we have

$$E_{\text{pot}} = Mg\left[\sin(\vartheta_{m,j})(p_{m,j}^{\text{end}} - x_{\text{init}}) + \sin(\vartheta_{m+1,j})(x_{\text{new}} - p_{m,j}^{\text{end}})\right] \tag{10}$$

Moreover, if the cybercar goes from the last segment of the current link to the first segment of the next link within the simulation interval, we get

$$E_{\text{pot}} = Mg\left[\sin(\vartheta_{m,j})(p_{m,j}^{\text{end}} - x_{\text{init}}) + \sin(\vartheta_{1,j^*})x_{\text{new}}\right] \tag{11}$$

By defining η_{motor} as the efficiency of electric motors[3], the actual energy consumption of the cybercar during the simulation time interval $[kT, (k+1)T]$ becomes

$$E(v_{\text{init}}, v_{\text{new}}, x_{\text{init}}, x_{\text{new}}, \vartheta_{m,j}, \vartheta_{m+1,j}, a_{\max}, T) = \max\left(\frac{E_{\text{total}}}{\eta_{\text{motor}}},\ 0\right) \tag{12}$$

where

$$E_{\text{total}} = E_{\text{var\_kin}} + E_{\text{var\_air}} + E_{\text{var\_rol}} + E_{\text{ct\_air}} + E_{\text{ct\_rol}} + E_{\text{pot}} \tag{13}$$

Moreover, if a cybercar uses regenerative braking, it could save part of the energy lost in braking to recharge its onboard battery. By letting γ_{recover} denote the round-trip energy recovery coefficient[4] of the regenerative braking system, we have

$$E(v_{\text{init}}, v_{\text{new}}, x_{\text{init}}, x_{\text{new}}, \vartheta_{m,j}, \vartheta_{m+1,j}, a_{\max}, T) = \begin{cases} \dfrac{E_{\text{total}}}{\eta_{\text{motor}}}, & \text{if } E_{\text{total}} \geq 0 \\ \dfrac{\gamma_{\text{recover}} \cdot E_{\text{total}}}{\eta_{\text{motor}}}, & \text{if } E_{\text{total}} < 0 \end{cases} \tag{14}$$

[3]According to [28], the maximal efficiency of electric motors is about 85% to 90%. That means for the best case, only 90% of the electricity consumed in charging the onboard battery can be used to power the electric vehicle.

[4]The round-trip energy recovery coefficient (i.e., the ratio between the amount of electric energy recovered from braking and the amount consumed in accelerating) of an electric vehicle is around 38% [29].

Finally, by defining $E_i(k)$ as the amount of energy consumed by cybercar i during $[kT,(k+1)T]$, we have

$$E_i(k) = E\Big(v_i(k), v_i(k+1), x_i(k), x_i(k+1), \vartheta_{s_i(k),l_i(k)}, \vartheta_{s_i(k+1),l_i(k+1)}, a_{\max,i}, T\Big) \quad (15)$$

IV. MODEL PREDICTIVE DYNAMIC ROUTING

Model predictive control (MPC) has been widely recognized as a high-performance control approach for complex and constrained systems [30], [31]. In MPC, the control actions over a certain time span in the future are determined by solving a constrained optimization problem that includes the model of the system, the operational constraints, and the goal of control explicitly, in a receding horizon fashion. Since the dynamics of cybercars are highly complex and subject to many constraints, based on the discrete-time model presented in Section III, we adopt an MPC scheme to formulate the dynamic routing problem of cybercars.

A. Overall control problem

During $[kT,(k+N_\mathrm{p})T]$, the total time spent and the total energy consumption by all cybercars are given by

$$J_\mathrm{TTS}(k) = \sum_{i\in I(k,N_\mathrm{p})} \min\Big((k+N_\mathrm{p})T - T_{\mathrm{start},i},\ T_{\mathrm{stop},i} - kT,\ T_{\mathrm{stop},i} - T_{\mathrm{start},i},\ N_\mathrm{p}T\Big) + J^\mathrm{end}_\mathrm{TTS}(k) \quad (16)$$

$$J_\mathrm{TEC}(k) = \sum_{h=1}^{N_\mathrm{p}} \sum_{i\in I(k,N_\mathrm{p})} E_i(k+h) + J^\mathrm{end}_\mathrm{TEC}(k) \quad (17)$$

where N_p denotes the prediction horizon and $I(k,N_\mathrm{p})$ denotes the set of cybercars in the network during $[kT,(k+N_\mathrm{p})T]$, $J^\mathrm{end}_\mathrm{TTS}(k)$ and $J^\mathrm{end}_\mathrm{TEC}(k)$ denote respectively estimates of the expected remaining total time spent and the expected remaining total energy consumption by the cybercars still in the network at $t=(k+N_\mathrm{p})T$ from their positions at $t=(k+N_\mathrm{p})T$ to their destinations. One possible way to obtain $J^\mathrm{end}_\mathrm{TTS}(k)$ and $J^\mathrm{end}_\mathrm{TEC}(k)$ is by using the speeds of the cybercars still in the network at $t=(k+N_\mathrm{p})T$ and considering the shortest time routes for those cybercars computed using e.g. *Dijkstra's algorithm* [32] based on their speeds at $t=(k+N_\mathrm{p})T$.

In order to properly balance $J_\mathrm{TTS}(k)$ and $J_\mathrm{TEC}(k)$, who have possibly different units and different orders of magnitude, the overall objective function is designed as

$$J(k) = w_1 \frac{J_\mathrm{TTS}(k)}{J_\mathrm{TTS,typical}} + w_2 \frac{J_\mathrm{TEC}(k)}{J_\mathrm{TEC,typical}} \quad (18)$$

where $J_\mathrm{TTS,typical}$ and $J_\mathrm{TEC,typical}$ denote typical values[5] of the total time spent and the total energy consumption by all cybercars in one prediction period while w_1, w_2 are nonnegative weights. Note that the control variable $r_i(k)$ for each cybercar i with $i\in I(k,N_\mathrm{p})$ is the route to be selected from a finite set $R_i(k)$ of possible routes[6] from its current position to its destination. Once the route $r_i(k)\in R_i(k)$ is determined, the link sequence following $r_i(k)$ can be determined and then used as input for the model presented Section III.

Since the dynamics of cybercars are nonlinear and the control variables are discrete, this results in a *Nonlinear Integer Programming* problem. Although there are several algorithms, such as genetic algorithm [34], simulated annealing [34] and DIRECT [35], available for solving this problem, in general, this problem is computationally very hard to solve, especially when the number of cybercars is large. For reasons of scalability and fast computation, a major challenge of achieving dynamic routing for a large fleet of cybercars is to find efficient approximate solution methods. In the next two sections, we propose efficient approximate solution methods for the dynamic routing of cybercars.

V. MULTI-AGENT MODEL PREDICTIVE DYNAMIC ROUTING

In this section, we propose a multi-agent model predictive control method for the dynamic routing of cybercars according to standard multi-agent MPC methods from literature. Given the similarities between the cybernetic transportation network considered in this paper and the transportation networks considered in [18], we adapt the parallel multi-agent MPC scheme presented in [18].

A. Decomposing the overall network

The whole network is divided into a set G of subnetworks[7]. To each subnetwork, an agent is assigned. At every control step, for each cybercar i, the sequence of subnetworks that will be visited by a cybercar i is first extracted from the shortest-time route from its current position to its destination computed by a shortest path algorithm based on the current traffic condition. After that, the exact route of the cybercar through each subnetwork is determined by the corresponding agent.

For each subnetwork g with a set of neighboring subnetworks $P_g=\{p_{g,1},...,p_{g,n_g}\}$, we define:

- $X_g(k)$: local state at time kT including positions, speeds, links and segments of cybercars in the local subnetwork g as well as the traffic densities and the blocking signals of all the links in the local subnetwork
- $U_g(k)$: local control variables i.e., routes of cybercars in the local subnetwork g
- $\omega_{\mathrm{in},g}(k) = [\omega^\mathrm{T}_{\mathrm{in},p_{g,1},g}(k),...,\omega^\mathrm{T}_{\mathrm{in},p_{g,n_g},g}(k)]^\mathrm{T}$: external inputs from neighboring subnetworks $p_{g,1},...p_{g,n_g}$ including the indices, entering points, entering times of the cybercars from neighboring subnetworks to the local subnetwork
- $\omega_{\mathrm{out},g}(k) = [\omega^\mathrm{T}_{\mathrm{out},p_{g,1},g}(k),...,\omega^\mathrm{T}_{\mathrm{out},p_{g,n_g},g}(k)]^\mathrm{T}$: outputs to neighboring subnetworks including the indices, exit

[5]These values could e.g., be the averaged total time spent and the averaged total energy consumption of cybercars over all prediction periods in a simulation where the routes of cybercars are predefined or a simple routing strategy (e.g., shortest time route) is used.

[6]This set of routes could be obtained using a K-shortest path algorithm [33] to find a fixed a number of possible routes from the end of the current link of cybercar i to its destination.

[7]Note that dividing a network into subnetworks, for which an efficient algorithm has been proposed by [36], is outside the scope of our work. We assume that the network and its division are given, and we only focus on the design of the multi-agent model predictive dynamic routing method.

points, and exit times of the cybercars from the local subnetwork to its neighboring subnetworks

B. MPC of a single subnetwork

We assume that at time kT, agent g has full knowledge of the current state of its own subnetwork and of the cybercars in its own subnetwork. Then, given the external inputs $\omega_{\text{in},g}(k)$ from neighboring subnetworks, agent g predicts the future local states using a local model. Given the predicted future local states, we define the following local objective function for agent g at time kT:

$$\begin{aligned} J_{\text{TTS},g}(k) = \sum_{i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)} \min\Big((k+N_\text{p})T - T_{\text{start},g,i},\ N_\text{p}T, \\ T_{\text{stop},i} - T_{\text{start},g,i},\ T_{\text{stop},i} - kT,\ T_{\text{cross},g,i} - T_{\text{start},g,i}, \\ T_{\text{cross},g,i} - kT\Big) + J^{\text{leave}}_{\text{TTS},g}(k) \end{aligned} \tag{19}$$

$$J_{\text{TEC},g}(k) = \sum_{h=1}^{N_\text{p}} \sum_{i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)} E_i(k+h) + J^{\text{leave}}_{\text{TEC},g}(k) \tag{20}$$

$$J_g(k) = w_1 \frac{J_{\text{TTS},g}(k)}{J_{\text{TTS,typical}}} + w_2 \frac{J_{\text{TEC},g}(k)}{J_{\text{TEC,typical}}} \tag{21}$$

where $\Omega_g(k)$ denotes the set of cybercars in subnetwork g at time kT, $\Omega_{g,\text{in}}(k)$ denotes the set of cybercars entering subnetwork g from neighboring subnetworks during $[kT,(k+N_\text{p})T]$. More specifically, $\Omega_{g,\text{in}}(k)$ is extracted from $\omega_{\text{in},g}(k)$. Furthermore, $T_{\text{cross},g,i}$ denotes the time when car i leaves the subnetwork of agent g and enters another subnetwork, $J^{\text{leave}}_{\text{TTS},g}(k)$ and $J^{\text{leave}}_{\text{TEC},g}(k)$ are estimates of the expected remaining total time spent and the expected remaining total energy consumption by the cybercars still in subnetwork g at time $(k+N_\text{p})T$, from $(k+N_\text{p})T$ to the time they leave the local subnetwork.

Finally, the following local control problem is solved by agent g:

$$\min_{\{r_i(k)|i\in\Omega_g(k)\cup\Omega_{g,\text{in}}(k)\}} J_g(k) \tag{22}$$

s.t.

$$r_i(k) \in R_{i,g}(k) \tag{23}$$

where $R_{i,g}(k)$ is a finite set of possible routes for cybercar i to go through subnetwork g.

C. Multi-agent model predictive dynamic routing method

Considering the interconnections among subnetworks and given the formulation of the local control problem of every subnetwork, we adapt the parallel multi-agent MPC scheme presented in [18] and apply it to the dynamic routing of cybercars. More specifically, the interconnecting constraints among subnetworks are removed from the constraint set and added to the objective function in the form of additional penalties based on an augmented Lagrangian formulation of the overall control problem. By using such an approach, the formulated problem becomes separable and can then be distributed over the agents. At each control step, the agents solve their local problems iteratively for fixed Lagrange multipliers, followed by updating the Lagrange multipliers using local solutions. The iterations stop when the Lagrange multipliers do not change anymore or the maximum allowed number of iterations is reached. After that, the agents implement the control actions until the next control step, after which the whole procedure is repeated.

VI. PARAMETERIZED DYNAMIC ROUTING

Determining the routes for all cybercars by solving an online optimization problem requires a huge computational effort. Therefore, in order to obtain a balanced trade-off between control performance and computational effort, we propose the use of parameterized control methods. The main idea of parameterized control is to parameterize the control decision-making process and to optimize the parameters of the control law by solving an optimization problem considering the performance of the control method, see [20] and [21]. After that, the control input are determined by using the parameterized control method with the optimized parameters.

In parameterized dynamic routing of cybercars, the selection process for cybercars is described using a parameterized control law that is a function of the state of the network. The parameters are optimized so as to optimize the routing performance including the total time spent and the total energy consumption of cybercars. After that, at each control cycle, the route of each cybercar is updated by selecting a route from a limited set of possible routes from its current position to its destination. Note that in different subnetworks, different values of the parameters or even different parameterized control laws may be used. For each parameterized dynamic routing method in this paper, we consider that the same parameterized control law is used but with different values of the parameters in different subnetworks.

At any time, a finite set of possible routes for each cybercar from its current position to its destination can be generated by using the current state of the network and by using shortest-route algorithms. More specifically, before a shortest-route algorithm is called to generate the limited sets of possible routes for cybercars, the estimated cost on each link j based on the current state of the network is determined by:

$$c_j = \lambda_1 \frac{L_{\text{link},j}}{L_{\text{link,ave}}} + \lambda_2 \frac{1}{T_{\text{link,ave}}} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k)\big)} \tag{24}$$

where $L_{\text{link},j}$ denotes the length of link j, $L_{\text{link,ave}}$ denotes the average of $L_{\text{link},j}$ over all links, $T_{\text{link,ave}}$ denotes the average link travel time over all links, $M_{\text{segment}}(j)$ denotes the number of segments in link j, λ_1 and λ_2 are given constants. One way to determine $T_{\text{link,ave}}$ is given by:

$$\bar{v}_{\text{free}} = \frac{\sum_{j=1}^{M_{\text{link}}} \sum_{m=1}^{M_{\text{segment}}(j)} v_{\text{free},m,j}}{\sum_{j=1}^{M_{\text{link}}} M_{\text{segment}}(j)} \tag{25}$$

$$T_{\text{link,ave}} = \frac{L_{\text{link,ave}}}{\gamma \cdot \bar{v}_{\text{free}}} \tag{26}$$

where M_{link} denotes the number of links in the network, $\bar{v}_{\text{free}}$ represents the average free-flow speed over all segments in all links, and γ is a model parameter.

A. Parameterized control method 1

We define $R_i(k)$ as the limited set of possible routes of cybercar i generated at time kT. After that, for each $r \in R_i(k)$, we define $L_{\text{route}}(r)$ as the length of route r, $T_{\text{route}}(r,k)$ as the estimated travel time on route r, and $N_{\text{route}}(r,k)$ as the estimated number of cybercars on route r.

Since the length of each link is fixed, the length of the route r can be easily calculated by summing up of the lengths of all the links belonging to route r. However, even if a route r is given, the travel time and the number of cybercars on that route are still time-dependent. Therefore, at any time when $T_{\text{route}}(r,k)$ and $N_{\text{route}}(r,k)$ are used, they have to be calculated based on the current states of all cybercars and of the network.

In this paper, we propose three approaches to estimate $T_{\text{route}}(r,k)$ and $N_{\text{route}}(r,k)$:

- Approach 1: Only use the current state of the network:

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k)\big)} \tag{27}$$

$$N_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} N_{m,j}(k) \tag{28}$$

- Approach 2: Predict the future states of the network assuming all cybercars follow the current routes and using the simulation model:

$$\bar{\rho}_{m,j}(k) = \sum_{l=1}^{N_{\text{p}}} \frac{\rho_{m,j}(k+l)}{N_{\text{p}}} \tag{29}$$

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\bar{\rho}_{m,j}(k)\big)} \tag{30}$$

$$N_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=1}^{N_{\text{p}}} \frac{N_{m,j}(k+l)}{N_{\text{p}}} \tag{31}$$

  where $\bar{\rho}_{m,j}(k)$ denotes the average traffic density in segment m of link j over $[kT,(k+N_{\text{p}})T]$.
- Approach 3: Predict the future states of the network assuming all cybercars follow the current routes. In this approach, $N_{\text{route}}(r,k)$ is estimated in the same way as in approach 2. However, different from approach 2, in this approach, $T_{\text{route}}(r,k)$ is estimated by

$$T_{\text{route}}(r,k) = \sum_{j \in r} \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=1}^{N_{\text{p}}} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k+l)\big)} \frac{1}{N_{\text{p}}} \tag{32}$$

Next, at time step k, for each cybercar i in subnetwork $g \in G$, we define a function for each $r \in R_i(k)$:

$$\begin{aligned}\varphi_i(r,\theta_g,k) = \; & \theta_{g,1} \cdot \frac{L_{\text{route}}(r,k)}{L_{\text{route,ave},i}(k)} + \theta_{g,2} \cdot \frac{T_{\text{route}}(r,k)}{T_{\text{route,ave},i}(k)} + \\ & + \theta_{g,3} \cdot \frac{N_{\text{route}}(r,k)}{N_{\text{route,ave},i}(k) + \kappa}\end{aligned} \tag{33}$$

where $\theta_{g,1}$, $\theta_{g,2}$, and $\theta_{g,3}$ are the parameters for subnetwork g and $L_{\text{route,ave},i}(k)$, $T_{\text{route,ave},i}(k)$, and $N_{\text{route,ave},i}(k)$ are respectively the average of $L_{\text{route}}(r,k)$, $T_{\text{route}}(r,k)$, and $N_{\text{route}}(r,k)$ over all $r \in R_i(k)$ for cybercar i, and κ is a small positive number added to the denominator to prevent division by 0. The route of each cybercar i in subnetwork g at kT is then selected as

$$r_i^* = \arg\min_{r \in R_i(k)} \varphi_i(r,\theta_g,k) \tag{34}$$

where $\theta_g = \begin{bmatrix} \theta_{g,1} & \theta_{g,2} & \theta_{g,3} \end{bmatrix}^{\text{T}}$.

B. Parameterized control method 2

In this method, we first define H_n as the set of outgoing links from node n and $R_{n,d}(k)$ as the limited set of possible routes from node n to node d generated at time kT. After that, for each $j \in H_n$, we define L_j as the length of link j, and $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$ as the estimated travel time and estimated number of cybercars on link j at time kT, respectively.

We propose three approaches to estimate $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$. More specifically, at time step k, given the current states of all cybercars and the current conditions of the network, $T_{\text{link}}(j,k)$ and $N_{\text{link}}(j,k)$ are estimated as follows:

- Approach 1:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k)\big)} \tag{35}$$

$$N_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} N_{m,j}(k) \tag{36}$$

- Approach 2:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \frac{L_{m,j}}{V_{m,j}\big(\bar{\rho}_{m,j}(k)\big)} \tag{37}$$

$$N_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=1}^{N_{\text{p}}} \frac{N_{m,j}(k+l)}{N_{\text{p}}} \tag{38}$$

- Approach 3:

$$T_{\text{link}}(j,k) = \sum_{m=1}^{M_{\text{segment}}(j)} \sum_{l=1}^{N_{\text{p}}} \frac{L_{m,j}}{V_{m,j}\big(\rho_{m,j}(k+l)\big)} \frac{1}{N_{\text{p}}} \tag{39}$$

Next, for each $j \in H_n$, we define $\tilde{r}_{j,d_i}$ as the shortest-time route from the end of link j to the destination node d_i. After that, for each cybercar i in an incoming link of node n with the link in subnetwork g, we define the following function:

$$\begin{aligned}\varphi_{i,n}(j,\theta_g,k) = \; & \theta_{g,1} \cdot \frac{L_{\text{link},j} + L_{\text{route}}(\tilde{r}_{j,d_i},k)}{L_{\text{ave},n,d_i}(k)} \\ & + \theta_{g,2} \cdot \frac{T_{\text{link}}(j,k) + T_{\text{route}}(\tilde{r}_{j,d_i},k)}{T_{\text{ave},n,d_i}(k)} \\ & + \theta_{g,3} \cdot \frac{N_{\text{link}}(j,k) + N_{\text{route}}(\tilde{r}_{j,d_i},k)}{N_{\text{ave},n,d_i(k)} + \kappa}\end{aligned} \tag{40}$$

where d_i denotes the destination of cybercar i, $\theta_{g,1}$, $\theta_{g,2}$, and $\theta_{g,3}$ are parameters for subnetwork g, and L_{ave,n,d_i}, T_{ave,n,d_i}, and N_{ave,n,d_i} are respectively the average of $L_{\text{route}}(r,k)$, $T_{\text{route}}(r,k)$, and $N_{\text{route}}(r,k)$ over all $r \in R_{n,d_i}(k)$.

Finally, the route of each cybercar i in an incoming link of node n and in subnetwork g is selected as follows:

- select the outgoing link from node n as

$$j^* = \arg\min_{j \in H_n} \varphi_{i,n}\left(j, \theta_g, k\right) \tag{41}$$

where $\theta_g = \begin{bmatrix} \theta_{g,1} & \theta_{g,2} & \theta_{g,3} \end{bmatrix}^{\mathrm{T}}$.

- the entire route of cybercar i from node n to its destination d_i is selected as:

$$r_i^* = \{j^*\} \cup \tilde{r}_{j^*,d_i} \tag{42}$$

C. Parameterized control method 3

Extended from method 1, parameterized control method 3 determines the route for each cybercar in a sequential way with updated network information (i.e., updated travel time and updated number of cybercars on a route) taking the updated routes of cybercars in the subnetwork into account.

In this method, we define $M_{n,d}(k)$ as the number of cybercars in the incoming links of node n at step k and heading to destination d, and $S_{n,d}(k)$ as the ordered set of the cybercars ordered according to their predicted arrival times at node n (e.g, based on their current speeds and the distance from their current positions to node n). After that, the $M_{n,d}(k)$ cybercars for every n and every d update their routes in the following sequential way:

i) for each $r \in R_{n,d}(k)$, calculate $L_{\mathrm{route}}(r,k)$, $T_{\mathrm{route}}(r,k)$ and $N_{\mathrm{route}}(r,k)$.

ii) Let $z = 1$

a) for the z-th cybercar in $S_{n,d}(k)$, update it route by

$$r_{i(z)}^* = \arg\min_{r \in R_{n,d}(k)} \varphi_{i(z)}\left(r, \theta_g, k\right)$$

where $i(z)$ is the global cybercar index that corresponds to the z-th cybercar in $S_{n,d}(k)$.

b) update $T_{\mathrm{route}}\left(r_{i(z)}^*, k\right)$ and $N_{\mathrm{route}}\left(r_{i(z)}^*, k\right)$ by

$$T_{\mathrm{route}}\left(r_{i(z)}^*, k\right) \leftarrow T_{\mathrm{route}}\left(r_{i(z)}^*, k\right) \cdot \left(1 + \frac{1\ [\mathrm{m}]}{L_{\mathrm{route}}\left(r_{i(z)}^*\right)}\right)$$

$$N_{\mathrm{route}}\left(r^*, k\right) \leftarrow N_{\mathrm{route}}\left(r_{i(z)}^*, k\right) + 1$$

iii) if $z < M_{n,d}(k)$, update $z \leftarrow z + 1$ and go back to a); otherwise, stop the procedure.

Note that $\varphi_i(\cdot)$ in this method is the same as (33).

D. Parameterized control method 4

Also extended from method 1, parameterized control method 4 determines the splitting rates of the group of cybercars over a limited set of possible routes.

In this method, for all $r \in R_{n,d}(k)$, we first define $L_{n,d,\mathrm{route}}^{\max}(k)$ as the length of the longest route, $T_{n,d,\mathrm{route}}^{\max}(k)$ as the longest estimated travel time following a route, and $N_{n,d,\mathrm{route}}^{\max}(k)$ as the largest number of cybercars on a route. Then we define

$$\Delta L_{\mathrm{route}}(r,k) = L_{n,d,\mathrm{route}}^{\max}(k) - L_{\mathrm{route}}(r,k)$$
$$\Delta T_{\mathrm{route}}(r,k) = T_{n,d,\mathrm{route}}^{\max}(k) - T_{\mathrm{route}}(r,k)$$
$$\Delta N_{\mathrm{route}}(r,k) = N_{n,d,\mathrm{route}}^{\max}(k) - N_{\mathrm{route}}(r,k)$$

After that, we define a function $\phi_{n,d}(\cdot)$ as

$$\phi_{n,d}\left(r, \theta_g, k\right) = \theta_{g,1} \cdot \frac{\Delta L_{\mathrm{route}}(r,k)}{L_{\mathrm{ave},n,d}(k)} + \theta_{g,2} \cdot \frac{\Delta T_{\mathrm{route}}(r,k)}{T_{\mathrm{ave},n,d}(k)} + + \theta_{g,3} \cdot \frac{\Delta N_{\mathrm{route}}(r,k)}{N_{\mathrm{ave},n,d}(k)} \tag{43}$$

where $\theta_{g,1}$, $\theta_{g,2}$, and $\theta_{g,3}$ are parameters for subnetwork g.

Further, the percentage of the $M_{n,d}(k)$ cybercars choosing route $r \in R_{n,d}(k)$ is determined by

$$P_{n,d}\left(r, \theta_g, k\right) = \frac{\phi_{n,d}\left(r, \theta_g, k\right)}{\sum_{y \in R_{n,d}(k)} \phi_{n,d}\left(y, \theta_g, k\right)} \tag{44}$$

Finally, the routes of cybercars in $S_{n,d}(k)$ are updated as follows:

i) the first round$\left(P_{n,d}\left(r_{\mathrm{first}}, \theta_g\right) \cdot M_{n,d}(k)\right)$ cybercars in $S_{n,d}(k)$ select the first route r_{first} in $R_{n,d}(k)$.

ii) after that, the following round$\left(P_{n,d}\left(r_{\mathrm{second}}, \theta_g\right) \cdot M_{n,d}(k)\right)$ cybercars in $S_{n,d}(k)$ select the second route r_{second} in $R_{n,d}(k)$.

iii) ...

ii) the remaining cybercars in $S_{n,d}(k)$ select the last route r_{last} in $R_{n,d}(k)$.

Note that after $R_{n,d}(k)$ is generated by using (24) and by using a shortest route algorithm, all the routes in $R_{n,d}(k)$ are ordered in an increasing sequence based on their costs. Here, $\{r_{\mathrm{first}}, ..., r_{\mathrm{last}}\}$ is the explicit representation of $R_{n,d}(k)$.

E. Parameterized control method 5

Parameterized control method 5 is extended from method 2 as in the same way method 3 is extended from method 1.

In this method, the $M_{n,d}(k)$ cybercars for every n and every d update their routes in the following sequential way:

i) for each $j \in H_n$, calculate $L_{\mathrm{link}}(j,k)$ and $L_{\mathrm{route}}(\tilde{r}_{j,d_i},k)$, $T_{\mathrm{link}}(j,k)$ and $T_{\mathrm{route}}(\tilde{r}_{j,d_i},k)$, $N_{\mathrm{link}}(j,k)$ and $N_{\mathrm{route}}(\tilde{r}_{j,d_i},k)$.

ii) Let $z = 1$

a) for the z-th cybercar in $S_{n,d}(k)$, update its route by first selecting the outgoing link from node n as:

$$j^* = \arg\min_{j \in H_n} \varphi_{i(z),n}\left(j, \theta_g, k\right)$$

and then set the entire route $r_{i(z)}^* = \{j^*\} \cup \tilde{r}_{j^*,d_i}$.

b) update $T_{\mathrm{route}}\left(r_{i(z)}^*, k\right)$ and $N_{\mathrm{route}}\left(r_{i(z)}^*, k\right)$ by

$$T_{\mathrm{route}}\left(r_{i(z)}^*, k\right) \leftarrow T_{\mathrm{route}}\left(r_{i(z)}^*, k\right) \cdot \left(1 + \frac{1\ [\mathrm{m}]}{L_{\mathrm{route}}\left(r_{i(z)}^*\right)}\right)$$

$$N_{\mathrm{route}}\left(r_{i(z)}^*, k\right) \leftarrow N_{\mathrm{route}}\left(r_{i(z)}^*, k\right) + 1$$

iii) if $z < M_{n,d}(k)$, update $z \leftarrow z + 1$ and go back to a); otherwise, stop the procedure.

Note that $\varphi_{i,n}(\cdot)$ in this method is the same as (40).

F. Parameterized control method 6

Finally, parameterized control method 6 is extended from method 2 as in the same way method 4 is extended from method 1.

Based on the definition of $L^{\max}_{n,d,\text{route}}(k)$, $T^{\max}_{n,d,\text{route}}(k)$ and $N^{\max}_{n,d,\text{route}}(k)$ in method 4, for each $j \in H_n$, we first define

$$\begin{aligned}\psi_{n,d}(j,\theta_g,k) = \theta_{g,1}\cdot\frac{L^{\max}_{n,d,\text{route}}(k)-\left(L_{\text{link},j}+L_{\text{route}}(\tilde{r}_{j,d})\right)}{L_{\text{ave},n,d}} \\ +\theta_{g,2}\cdot\frac{T^{\max}_{n,d,\text{route}}(k)-\left(T_{\text{link}}(j,k)+T_{\text{route}}(\tilde{r}_{j,d})\right)}{T_{\text{ave},n,d}} \\ +\theta_{g,3}\cdot\frac{N^{\max}_{n,d,\text{route}}(k)-\left(N_{\text{link}}(j,k)+N_{\text{route}}(\tilde{r}_{j,d})\right)}{N_{\text{ave},n,d}}\end{aligned} \tag{45}$$

where $\theta_{g,1}$, $\theta_{g,2}$, and $\theta_{g,3}$ are parameters for subnetwork g. After that, the percentage of the $M_{n,d}(k)$ cybercars choosing route $\{j\}\cup\tilde{r}_{j,d}$ is determined by

$$P_{n,d}(j,\theta_g,k) = \frac{\psi_{n,d}(j,\theta_g,k)}{\sum_{y\in H_n}\psi_{n,d}(y,\theta_g,k)} \tag{46}$$

Finally, given $P_{n,d}(j,\theta_g,k)$ for all $j \in H_n$, all cybercars in $S_{n,d}(k)$ update their routes in the way same as in method 4.

G. Tuning the parameters for parameterized control methods

To tune the parameters of the proposed parameterized control methods, we proceed as follows. We define a scenario as a case where the transport service requests including the starting times, the origins and the destinations of all cybercars, are given. Then, the performance of a parameterized control method on a specific scenario of the dynamic routing of cybercars is evaluated by

$$J_o(\boldsymbol{\theta}) = w_1\cdot\frac{J_{\text{TTS},o}}{J_{\text{TTS,typical,scenario}}} + w_2\cdot\frac{J_{\text{TEC},o}}{J_{\text{TEC,typical,scenario}}} \tag{47}$$

where o is the index of the scenario, $\boldsymbol{\theta} = \begin{bmatrix}\theta_1^{\mathrm{T}} & \theta_2^{\mathrm{T}} & \dots\end{bmatrix}^{\mathrm{T}}$, $J_{\text{TTS},o}$ and $J_{\text{TEC},o}$ respectively denote the total time spent and the total energy consumption by all cybercars, $J_{\text{TTS,typical,scenario}}$ and $J_{\text{TEC,typical,scenario}}$ respectively denote the typical values[8] of the total time spent and the total energy consumption by all cybercars in a representative scenario.

Finally, given a number N^{scenario} of representative scenarios, the parameters $\boldsymbol{\theta}$ of the parameterized control method are tuned by minimizing the sum of $J_o(\boldsymbol{\theta})$ over the representative scenarios. More specifically, the parameters $\boldsymbol{\theta}$ are tuned by solving the following *nonlinear programming* problem:

$$\begin{aligned}\min_{\boldsymbol{\theta}} \quad & \sum_{o=1}^{N^{\text{scenario}}} J_o(\boldsymbol{\theta}) \\ \text{s.t.} \quad & \text{model equations}\end{aligned} \tag{48}$$

[8]These values are e.g., the values of total time spent and total energy consumption of all cybercars in a numerical simulation where the routes of all cybercars are fixed or a simple route control strategy (e.g., fastest route) is used.

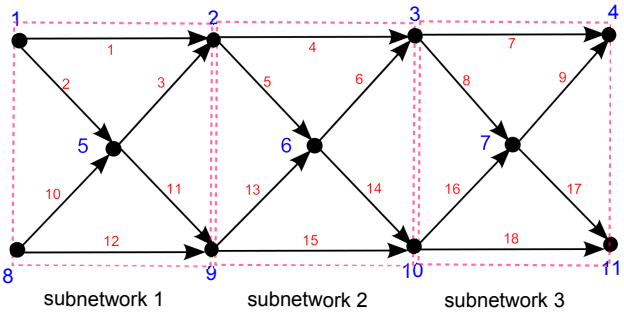

Fig. 3. Road network used in the case study

TABLE II
FLOW DIVISION OF CYBERCARS IN EVERY SCENARIO

Flows able to update routes			Flows with fixed routes		
Index	O - D	Percentage	Index	O - D	Percentage
1	5 - 7	40%	6	2 - 3	20%
2	1 - 10	15%	7	9 - 10	20%
3	8 - 3	15%	8	2 - 10	30%
4	2 - 11	15%	9	9 - 3	30%
5	9 - 4	15%			

which is nonconvex and can be solved by using multiple runs of nonconvex optimization algorithms, e.g. genetic algorithm, simulated annealing, pattern search, or sequential quadratic programming [37].

VII. SIMULATION STUDY

In this section, we perform simulation experiments to compare and assess the performance of the proposed control methods for dynamic routing of cybercars. We consider the network shown in Figure 3, where there are 11 nodes and 18 links. Each link is 200 meters long and has 4 segments, with each segment 50 meters long. In the simulations, we generated 30 scenarios, of which 20 are used for tuning the parameters of the proposed control methods and the other 10 are used for evaluating the performance of the proposed control methods. For every scenario o, we set a number $N_{\text{car,enabled}}$ of cybercars that are allowed to update routes and generate a random number $N_{\text{car,fixed}}$ of cybercars with fixed routes. This is done for the reason of considering more cybercars in the network without dramatically increasing the computational complexity of the control problem for centralized MPC and multi-agent MPC. More specifically, for every scenario o, $N_{\text{car,enabled}}$ is determined by

$$N_{\text{car,enabled},o} = 200 + 15\times\left(\left\lceil\frac{o}{3}\right\rceil - 1\right)$$

and $N_{\text{car,fixed}}$ is a random integer uniformly distributed in the interval $[100, 200]$. After $N_{\text{car,enabled}}$ and $N_{\text{car,fixed}}$ are set, the cybercars are divided into 9 flows, which are summarized in Table II. Note that for every three consecutive scenarios starting from $o = 1$, we use the first two for tuning the parameters of the proposed parameterized control methods and use the third one for testing the control performance.

For every scenario, we define the departure times of cybercars for each O-D flow as follows. For each of the flows 1, 2 and 3, the departure time of the first cybercar is a random

number uniformly distributed in $[0.8, 1.6]$ s. In order to create congestion we let the first cybercar of flows 4 and 5 depart later than that of flows 1, 2, and 3 by adding an offset of 40 s. So the first cybercar in each of the flows 4 and 5 departs at $(40+a)$ s, where a is a random number uniformly distributed in $[0.8, 1.6]$. For the subsequent cybercars in flows 1 to 5 the time interval between the departure times of two consecutive cybercars is a random number uniformly distributed in $[0.8, 1.6]$. Besides, for each flow of cybercars with fixed routes, the first cybercar departs at $(1.2+b)$ s, where is b a random number uniformly distributed in $[10, 20]$. After that, the time interval between the departure times of two consecutive cybercars is 1.2 s.

The other parameters used in the simulations are: $T = 1$s, $w_1 = 0.7$, $w_2 = 0.3$, $N_\mathrm{p} = 20$, $J_{\mathrm{TTS,typical,scenario}} = 73202$ s and $J_{\mathrm{TEC,typical,scenario}} = 11.68$ kWh, $L_\mathrm{veh} = 3.2$ m, $\eta_\mathrm{motor} = 0.85$, $\gamma_\mathrm{recover} = 0.38$, $v_{\mathrm{free},m,j} = 60$ km/h for all m and all j, the mass of each cybercar is $M = 1000$ kg, the time interval between two consecutive control steps is $T_\mathrm{c} = 20$ s. The simulations are performed using Matlab 2015a on a cluster computer consisting of 4 blades with 2 eight-core E5-2643 processors, and 3.3 GHz clock rate and 64 GiB memory per blade.

We tuned the parameters for the six proposed parameterized control methods with three different approaches for estimating the travel time and the number of cybercars on a route. For tuning the parameters of each of the 18 combinations, we run the solver *fmincon* of the Matlab Optimization Toolbox with the sequential quadratic programming (SQP) algorithm 60 times using random starting points to solve the *nonlinear programming* problem (48). For simplicity of representation, we refer to the six proposed parameterized dynamic routing methods with their three different estimation approaches as PC*x*-*y*, where x is the index of the parameterized dynamic routing method and y is the index of the estimation approach, e.g., PC4-3 presents the parameterized dynamic routing method 4 with estimation approach 3. The CPU times for tuning the parameters for the proposed parameterized control methods are in the range of $0.97 \cdot 10^5$ to $1.61 \cdot 10^5$ seconds.

After tuning the parameters, we evaluate the performance of the parameterized control methods on the 10 different testing scenarios. In order to show the effectiveness of the proposed parameterized control methods, we compare the performance of the proposed parameterized control methods on the testing scenarios with those of centralized model predictive control, multi-agent model predictive control, and three greedy control methods using *Dijkstra's Algorithm*. We refer to centralized model predictive control, multi-agent model predictive control, and greedy control methods in the following way:

- C-MPC: centralized MPC using multiple runs of the genetic algorithm with a limited total computation time (i.e., for all runs together) of 3600 s at each control step
- MA-MPC: multi-agent MPC with each agent solving its local problem using bilevel optimization with the following procedure:
  - fix the binary variables and next solve the problem to obtain the real decisions using *fmincon/SQP*
  - solve the binary optimization problem using the genetic algorithm with a limited total computation time of 3600 s at each control step
- GC1: greedy control method 1, i.e. shortest distance routing method, using *Dijkstra's algorithm* based on (24) with $\lambda_1 = 1$ and $\lambda_2 = 0$
- GC2: greedy control method 2, i.e. shortest time routing method, using *Dijkstra's algorithm* based on (24) with $\lambda_1 = 0$ and $\lambda_2 = 1$
- GC3: greedy control method 3, i.e. combined distance and time routing method, using *Dijkstra's algorithm* based on (24) with $\lambda_1 = 0.3$ and $\lambda_2 = 0.7$

TABLE III
AVERAGE ONLINE COMPUTATION TIMES (S) OF THE CONTROL METHODS

Control methods	Average online computation times (s)
Centralized MPC	69586
Multi-agent MPC	66502
Greedy control	0.07 - 0.14
Parameterized control	0.37 - 4.02

The average online computation times of all control methods over the testing scenarios are summarized in Table III. For the sake of compactness, only the ranges of average online computation times of the greedy control (GC) methods and the parameterized control (PC) methods are provided.

Since GC3 has the best performance among all the greedy control methods, we use GC3 as the benchmark and calculate the performance improvement of the other control methods for the testing scenarios. More specifically, the performance improvement of a routing method on a specific scenario o compared with that of GC3 on the same scenario is given by

$$P_{\mathrm{im},o} = \frac{J_{\mathrm{GC3},o} - J_o}{J_{\mathrm{GC3},o}} \times 100\%$$

The average performance improvement and the standard deviation of the performance improvement of the routing control methods compared with GC3 over all the testing scenarios are summarized in Table IV. Note that in the second column, a positive number in a cell indicates that the corresponding control method performs better than GC3 on the average over all the testing scenarios. We found that parameterized control method 4 with estimation approach 1 (i.e. PC4-1, the-flow-splitting-rate-based parameterized control method) has the best performance among all the proposed parameterized control methods and it has an average performance improvement of 5.04% on the testing scenarios compared with GC3. Therefore, PC4-1 is selected as the representative of the proposed parameterized control methods to compare further with centralized MPC, multi-agent MPC, and GC3 on all the testing scenarios. More specifically, Figure 4 shows the performance of centralized MPC, multi-agent MPC, GC3, and PC4-1 for all the testing scenarios.

Actually, we are performing state feedback routing control of cybercars with the proposed parameterized control methods. Different from the prediction of future state of the network in model predictive control where the decision variables are free, in estimation approach 2 and approach 3, the future state of the network are estimated assuming the routes of all

TABLE IV
AVERAGE PERFORMANCE IMPROVEMENT OF THE OTHER CONTROL METHODS WITH RESPECT TO GC3 FOR ALL TESTING SCENARIOS, WHERE A POSITIVE NUMBER INDICATES A BETTER PERFORMANCE

Control method	Performance improvement	Standard deviation
C-MPC	6.36%	4.4%
MA-MPC	1.45%	1.4%
GC1	-4.20%	3.7%
GC2	-0.71%	2.3%
PC1-1	2.67%	7.7%
PC1-2	4.46%	8.7%
PC1-3	3.45%	7.9%
PC2-1	1.99%	7.5%
PC2-2	2.92%	7.7%
PC2-3	3.29%	8.6%
PC3-1	3.89%	7.7%
PC3-2	2.86%	8.8%
PC3-3	4.26%	8.4%
PC4-1	5.04%	8.1%
PC4-2	3.46%	8.4%
PC4-3	2.95%	8.3%
PC5-1	1.45%	8.3%
PC5-2	3.24%	7.4%
PC5-3	3.18%	7.2%
PC6-1	3.90%	8.3%
PC6-2	1.87%	8.1%
PC6-3	1.36%	8.9%

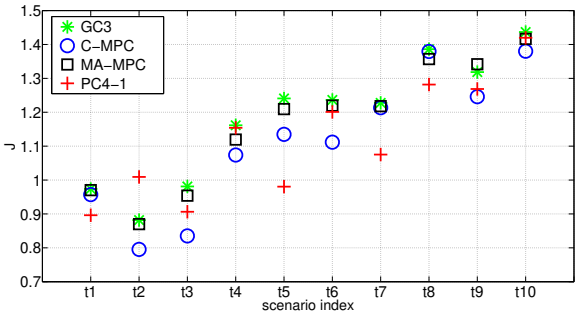

Fig. 4. Performance of GC3, centralized MPC, multi-agent MPC and PC4-1 for all testing scenarios

cybercars are fixed. If the updated routes of the cybercars are very different from the fixed ones for estimating the future state of the network, then the actual state of the network after the control step could be very different from the estimated states obtained by estimation approach 2 and approach 3. Therefore, even though only the current state of the network are used in estimation approach 1, there is no guarantee that estimation approach 2 and approach 3 perform better than estimation approach 1 in estimating the travel time and the number of cybercars on the routes in the network. Hence, there is no guarantee that a proposed parameterized control method with estimation approach 2 or approach 3 would have better performance in routing control of cybercars than that with estimation approach 1.

Further, it is seen from Table IV that the centralized MPC performs the best on the testing scenarios with an average performance improvement of 6.36% compared with GC3. However, this is achieved by consuming much more computational power, see Table III. For the multi-agent MPC, since the overall problem is a *mixed integer nonlinear programming* problem, there is no guarantee of convergence of interconnecting variables among subnetworks. In fact, even if given the same computational budget as that of the centralized MPC, the multi-agent MPC does not obtain a performance that is comparable to that of centralized MPC. In contrast, the proposed parameterized control method PC4-1 provides a comparable average performance to that of the centralized MPC on the testing scenarios with much less online computation time, also see Table III. For the standard deviations of the performance improvement compared with GC3, those of the centralized MPC and the multi-agent MPC are smaller than those of the parameterized control methods. That is because the centralized MPC and the multi-agent MPC use online optimization for every scenario while the parameters of the parameterized control methods are tuned based on representative scenarios and then fixed for online use.

Finally, it is seen from Figure 4 that PC4-1 performs better than GC3 on 9 of the 10 testing scenarios and performs better than the multi-agent MPC on 7 of the testing scenarios. Besides, the centralized MPC only performs better than PC4-1 on 6 of the testing scenarios while on the other 4 it is outperformed by PC4-1. Moreover, it has to be noted that the average online computation time of PC4-1 for all the testing scenarios is only 0.42 s. Therefore, the parameterized control method PC4-1 is an efficient method for the dynamic routing of a fleet of cybercars.

VIII. CONCLUSIONS

We have addressed the dynamic routing problem of a fleet of cybercars considering the dynamics and the energy consumption of every cybercar according to the real-time conditions of the road network. To minimize the total cost for all cybercars, i.e. a combination of the total time spent and the total energy consumption, we have developed tractable and scalable multi-agent control methods including multi-agent model predictive control and parameterized control. Numerical simulation results indicate that the flow splitting rate based parameterized control method shows comparable control performance to that of centralized model predictive control while requiring much less online computation time. Besides, the-flow-splitting-rate-based parameterized control method can be easily applied to road networks with arbitrary topology. Therefore, the flow-splitting-rate-based parameterized control method is effective in solving the dynamic routing problem of a fleet of cybercars.

In our future work, we will first focus on increasing the computation efficiency of the proposed dynamic routing methods by investigating different levels of model aggregation. We will also perform more detailed case studies for extensive assessment of the performance and the efficiency of the proposed multi-agent dynamic routing methods including more complex scenarios with larger-scale road networks. In addition, we will consider the scenarios where conventional vehicles and cybercars coexist in the network and address the joint dynamic routing of conventional vehicles and cybercars. Furthermore, we will validate the proposed model of the dynamics of cybercars and the proposed dynamic routing methods using actual data.

REFERENCES

[1] M. Parent and P. Texier. A public transport system based on light electric cars. In *Proceedings of 4th International Conference on Automated People Movers*, pages 154–161, Irving, USA, March 1993.

[2] M. Parent. Cybercars for sustainable urban mobility - A European collaborative approach. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 3(2):220–223, 2010.

[3] A. Awasthi, S. S. Chuanhan, M. Parent, and J. M. Proth. Centralized fleet management system for cybernetic transportation. *Expert Systems with Applications*, 38(4):3710–3717, 2011.

[4] M. Parent, G. Gallais, A. Alessandrini, and T. Chanard. Cybercars: Review of first projects. In *Proceedings of 9th International Conference on Automated People Movers*, Singapore, September 2003.

[5] M. Yang, C. Wang, R. Yang, and M. Parent. Cyberc3: Cybercars automated vehicles in China. In *Proceedings of Transportation Research Board Annual Meeting*, Washington, D.C., USA, January 2006.

[6] L. Vlacic, M. Parent, and F. Harashima. *Intelligent vehicle technologies: theory and applications*. Butterworth-Heinemann, 2001.

[7] C. Hatipoglu, U. Ozguner, and K. Redmill. Automated lane change controller design. *IEEE Transactions on Intelligent Transportation Systems*, 4(1):13–22, 2003.

[8] D. Corona and B. De Schutter. Adaptive cruise control for a smart car: A comparison benchmark for MPC-PWA control methods. *IEEE Transactions on Control Systems Technology*, 16(2):365–372, 2008.

[9] T. Berger, Y. Sallez, S. Raileanu, C. Tahon, D. Trentesaux, and T. Borangiu. Personal rapid transit in an open-control framework. *Computer & Industrial Engineering*, 61(2):300–312, 2011.

[10] T. Garaix, C. Artigues, D. Feillet, and D. Josselin. Vehicle routing problems with alternative paths: An application to on-demand transportation. *European Journal of Operational Research*, 204(1):62–75, 2010.

[11] A. Hegyi, B. De Schutter, and J. Hellendoorn. Optimal coordination of variable speed limits to suppress shock waves. *IEEE Transactions on Intelligent Transportation Systems*, 6(1):102–112, 2005.

[12] A. Kotsialos, I. Papamichail, I. Margonis, and M. Papageorgiou. Hierarchical nonlinear model-predictive ramp metering control for freeway networks. In *Proceedings of the 11th IFAC Symposium on Control in Transportation Systems*, pages 124–129, Delft, The Netherlands, 2006.

[13] J. R. D. Frejo and E. F. Camacho. Global versus local MPC algorithms in freeway traffic control with ramp metering and variable speed limits. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1556–1565, 2012.

[14] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar. Distributed model predictive control. *IEEE Control Systems Magazine*, 22(1):44–52, 2002.

[15] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, and S. J. Wright. Distributed MPC strategies with application to power system automatic generation control. *IEEE Transactions on Control Systems Technology*, 16(6):1192–1206, 2008.

[16] B. T. Stewart, A. N. Venkat, J. B. Rawlings, S. J. Wright, and G. Pannocchia. Cooperative distributed model predictive control. *Systems & Control Letters*, 59(8):460–469, 2010.

[17] J. M. Maestre, D. Muñoz de la Peña, and E. F. Camacho. Distributed model predictive control based on a cooperative game. *Optimal Control Applications and Methods*, 32(5):153–176, 2011.

[18] R. R. Negenborn, B. De Schutter, and J. Hellendoorn. Multi-agent model predictive control for transportation networks: Serial versus parallel schemes. *Engineering Applications of Artificial Intelligence*, 21(3):353–366, 2008.

[19] S. K. Zegeye, B. De Schutter, J. Hellendoorn, E. A. Breunesse, and A. Hegyi. A predictive traffic controller for sustainable mobility using parameterized control policies. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1420–1429, 2012.

[20] R. Oung, M. P. Cruz, and R. D'Andrea. A parameterized control methodology for a modular flying vehicle. In *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 532–538, Algarve, Portugal, 2012.

[21] A.N. Tarău, B. De Schutter, and J. Hellendoorn. Model-based control for route choice in automated baggage handling systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 40(3):341–351, 2010.

[22] R. Luo, T.J.J. van den Boom, and B. De Schutter. Modeling of the dynamics and the energy consumption of a fleet of cybercars. In *Proceedings of the 2014 European Control Conference*, pages 720–725, Strasbourg, France, June 2014.

[23] M. Treiber and A. Kesting. *Traffic Flow Dynamics: Data, Models and Simulation*. Springer-Verlag, 2010.

[24] D. Helbing. Derivation of a fundamental diagram for urban traffic flow. *The European Physical Journal B*, 70(2):229–241, 2009.

[25] S. Yagar and M. Van Aerde. Geometric and environmental effects on speeds of 2-lane highways. *Transportation Research Part A: General*, 17(4):315–325, 1983.

[26] A. Bose and P. Ioannou. Mixed manual/semi-automated traffic: a macroscopic analysis. *Transportation Research Part C: Emerging Technologies*, 11(6):439–462, 2003.

[27] G. Dervisoglu, G. Gomes, J. Kwon, R. Horowitz, and P. Varaiya. Automatic calibration of the fundamental diagram and empirical observations on capacity. In *Proceedings of Transportation Research Board Annual Meeting*, Washington, D.C., USA, January 2009.

[28] D. MacKay. *Sustainable Energy-Without the Hot Air*. UIT Cambridge, 2008.

[29] L. Rambaldi, E. Bocci, and F. Orecchini. Preliminary experimental evaluation of a four wheel motors, batteries plus ultracapacitors and series hybrid powertrain. *Applied Energy*, 88(2):442–448, 2011.

[30] E. F. Camacho and C. Bordons. *Model Predictive Control in Process Industry*. Springer-Verlag, Berlin, Germany, 1995.

[31] J. M. Maciejowski. *Predictive Control with Constraints*. Prentice-Hall, Harlow, England, 2002.

[32] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[33] D. Eppstein. Finding the K shortest paths. *SIAM Journal on Computing*, 28(2):652–673, 1998.

[34] P. M. Pardalos and M. G. C. Resende, editors. *Handbook of Applied Optimization*. Oxford University Press, Oxford, UK, 2002.

[35] D. R. Jones. DIRECT global optimization algorithm. In *Encyclopedia of Optimization*, pages 431–440. Springer, 2001.

[36] C. A. Farrell, D. H. Kieronska, and M. Schulze. Genetic algorithms for network division problem. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation*, pages 422–427, 1994.

[37] D. P. Bertsekas. Nonlinear Programming. *Athena Scientific*, 1999.

Renshi Luo received his Ph.D degree in Systems and Control from the Delft Center for Systems and Control, Delft University of Technology, The Netherlands, in 2016. Currently, he is a Postdoctoral Associate at the Hybrid System Control Team of CentraleSupelec, France.

His research interests include multi-agent control, distributed and large-scale systems, hybrid systems, and intelligent transportation and infrastructure systems.

Ton J.J. van den Boom received his PhD degree in Electrical Engineering from Eindhoven University of Technology, The Netherlands, in 1993. Currently, he is an Associate Professor at the Delft Center for Systems and Control of Delft University of Technology, The Netherlands.

His research interests are in the areas of linear and nonlinear model predictive control, and control discrete-event and hybrid systems.

Bart De Schutter (IEEE member since 2008, senior member since 2010) received the PhD degree in 1996, at K.U.Leuven, Belgium. Currently, he is a full professor at the Delft Center for Systems and Control of Delft University of Technology in Delft, The Netherlands. Bart De Schutter is senior editor of the IEEE Transactions on Intelligent Transportation Systems and associate editor of Automatica.

His current research interests include intelligent transportation and infrastructure systems, discrete-event systems, hybrid systems, and multi-level and distributed control.