Eco-Driving in Urban Traffic Networks Using Traffic Signals Information

Giovanni De Nunzio, Carlos Canudas de Wit, Philippe Moulin, Domenico Di Domenico

Eco-Driving in Urban Traffic Networks Using Traffic Signals Information

Giovanni De Nunzio*[1,2], Carlos Canudas de Wit[2], Philippe Moulin[1], and Domenico Di Domenico[1]

[1] *IFP Energies Nouvelles, Department of Control Signal and Systems, 1&4 avenue de Bois-Préau, 92852, Rueil-Malmaison, France.*

[2] *NeCS team, Inria Grenoble Rhône-Alpes, 655 Avenue de l'Europe, 38334 Montbonnot Saint-Martin, France.*

Abstract

The problem of eco-driving is analyzed for an urban traffic network in presence of signalized intersections. It is assumed that the traffic lights timings are known and available to the vehicles via infrastructure-to-vehicle (I2V) communication. This work provides a solution to the energy consumption minimization, while traveling through a sequence of signalized intersections and always catching a green light. The optimal control problem is non-convex due to the constraints coming from the traffic lights, therefore a sub-optimal strategy to restore the convexity and solve the problem is proposed. Firstly, a pruning algorithm aims at reducing the optimization domain, by considering only the portions of the traffic lights green phases that allow to drive in compliance with the city speed limits. Then, a graph is created in the feasible region, in order to approximate the energy consumption associated with each available path in the driving horizon. Lastly, after the problem convexity is recovered, a simple optimization problem is solved on the selected path to calculate the optimal crossing times at each intersection. The optimal speeds are then suggested to the driver. The proposed sub-optimal strategy is compared to the optimal solution provided by Dynamic Programming, for validation purposes. It is also shown that the low computational load of the presented approach enables robustness properties, and results very appealing for online use.

***Keywords* —** Eco-driving, nonlinear optimization, speed advisory, trajectory planning.

1 Introduction

Pollution reduction and energy consumption optimization are becoming more and more critical. Over the last years the number of vehicles demanding access to the road infrastructure has increased exponentially. Governments along with scientific community aim at reducing both environmental impact and costs of transportation. In particular the European Union has set binding legislation to cut emissions by 20% with respect to 1990 levels by the end of

*IFPen, 1&4 avenue de Bois-Préau, 92852, Rueil-Malmaison, France. E-mail: giovanni.de-nunzio@ifpen.fr

2020 [1]. Traffic congestion and idling time at signalized intersections are among the main causes of energy consumption. Congestion in 2010 has been calculated to cost to US drivers about 101 billions of dollars [2]. Transportation is responsible for a substantial fraction of total green-house gas emissions (GHG), around a quarter in the European Union, making it the second biggest GHG emitting sector after energy production. However, while emissions from other sectors are generally decreasing, those from transportation have increased since 1990 (by 36% in the European Union). In [3], experiments showed that the concentration of fine particles in the air at traffic lights during stops is 29-times higher as compared to free-flow conditions. Also, though the delay time at intersections represents only a minor portion of the entire commuting time, it may contribute up to about 25% of the total trip emissions.

Extensive study and experimentation of several adaptive traffic control systems have been conducted over the past three decades. Strategies such as SCOOT [4], which reduces traffic delay by about 20% in urban areas, SCATS and TUC [5] have been employed all over the world proving actual benefits in terms of congestion and emissions reduction. However, these strategies present some limitations in terms of traffic conditions responsiveness and sensor failure robustness.

The state of the art in wireless communication, the deployment of dedicated communication protocols for Vehicular Ad-hoc Networks (VANETs), and the decreasing price of GPS receivers, allow more and more to rely on communication between the different agents of an urban traffic network for the design of robust and efficient traffic control strategies [6]. Specifically, Infrastructure-to-Vehicle (I2V) and Vehicle-to-Vehicle (V2V) communication attracted the attention of many due to their potential to enable fast and cheap advanced driving assistance systems (ADAS). Several international projects and groups (Drive C2X, iTetris, COMeSafety2, PATH), involving both automotive manufacturers and research centers, have been working on the setup of the communication infrastructure and on the assessment of the effect of this technology on traffic management and energy consumption.

Speed advisory for the urban environment was already proposed in the '80s [7][8] as a very energy-efficient traffic management strategy and as a pioneer for the modern ADAS. The rather simple initial idea of placing a roadside sign upstream from an intersection, indicating the travel speed to catch a green light, nowadays can be substantially improved thanks to the road communication networks.

Knowledge of the traffic lights signal timings has been proved to be significantly beneficial in terms of energy efficiency in urban traffic. Green lights optimal speed advisory (GLOSA), with different penetration rate of the technology, has shown promising advantages in terms of fuel consumption and average idling time [9]. Vehicles energy consumption is strictly related to the accelerations in driving profiles. Soft decelerations to traffic lights on red, using advanced wireless-transmitted information, have been proved to be more energy efficient than sudden stops, allowing a reduction of both fuel consumption and emissions [10]. In general, it is possible to reduce energy consumption by preventing a vehicle from coming to a full stop at the intersections and by advising cruising velocities in order to catch as many green lights as possible. This goal could be achieved in different ways. In [11], authors developed an algorithm to minimize the acceleration profile while driving through a sequence of signalized intersections, simulating scenarios with stochastically varied parameters and showing a systematic reduction of fuel consumption, emissions and travel time. Interesting results on eco-driving with probabilistic signal phase and timing (SPAT) information are shown in [12], where a comparison between an uninformed driver and a driver with different information levels (full horizon and short-term) is conducted using Dynamic

Programming (DP). An innovative solution to the velocity advisory problem, in proximity to a traffic light, was proposed in [13], where smartphones make use of their cameras to build a SPAT map of every intersection by relying only on V2V communication, without directly interacting with the infrastructure. This approach may enable GLOSA service and adaptive route change (ARC). Predictive cruise control (PCC) in traffic networks with signalized intersections was extensively treated in [14]. The objective is to penalize the braking action to indirectly reduce energy consumption and travel time, and to discourage deviations from a suggested speed that allows to cross intersections without stopping. Somewhat similar approach was used in [15]. A preliminary algorithm determines the arrival times at each traffic light, by assuming that the closest trajectory to the one of the unconstrained case (i.e. no traffic lights) is the most energy-efficient. Then an analytical and numerical solution to the fuel consumption minimization problem is proposed, for a scenario with three traffic lights over a driving horizon of nine kilometers. In [16] only one traffic light is considered and accelerations upstream and downstream from the intersection are varied to find sub-optimal values for the fuel consumption minimization problem. In [17] DP is used to find the minimum energy solution, for a driving horizon enclosing up to two traffic lights. An approach to the multi-segment speed advisory is presented in [18]. The fuel consumption is indirectly minimized by penalizing speed variations between adjacent road sections. The strategy accounts for multiple intersections and shows how such optimization yields better results with respect to the single-segment approach. However nothing is stated about the computation time and online applicability of the algorithm.

Assuming I2V communication, and therefore full knowledge of the traffic lights timings, the goal of this work is to analyze the driving horizon and compute an energy-efficient speed advisory for the driver. As in previous works, stops at a red traffic light are to be avoided. The novelty of this work is summarized as follows. Given a set of green traffic light phases, there exist different driving profiles to reach a given destination at a given final time in compliance with traffic lights constraints (i.e. always catching the green light) and city speed limits. The presented strategy is capable of an a priori identification of the most energy-efficient velocity trajectory, by approximating the available paths and their energy cost with an oriented weighted graph. The computational complexity of the graph creation has been reduced in this work from exponential [19] to polynomial, thanks to the introduction of the line graph. The computation time has been consequently significantly reduced. Only after this preliminary stage of path selection, a formal optimization problem is solved in order to calculate the optimal arrival times at each intersection, by explicitly minimizing the energy consumption of the vehicle. This approach qualifies as a pre-trip eco-driving ADAS, since the speed advisory is provided to the driver at the beginning of the driver horizon. However, given the very little computation time required by the algorithm, it may be employed online thus enabling in-trip assistance features. This allows to respond dynamically to traffic perturbations and/or deviations from the speed advisory, and to increase the robustness and the applicability of the strategy in a realistic environment. Simulations in a microscopic traffic simulator demonstrate that the proposed strategy is able to deal online with perturbations coming from traffic and to reduce the overall energy consumption without affecting travel time.

Section 2 introduces the vehicle model and formulates the optimization problem. In Section 3 the applied methodology for the sub-optimal solution of the problem is explained in detail. In Section 4 the algorithm is validated against the true energy consumption calculated via DP, and robustness capabilities are shown. Concluding remarks are given in Section 5.

2 Problem Formulation

This work has as objective the minimization of the energy consumed by vehicles in a traffic network to travel from an initial to a final point, in presence of signalized intersections in the driving horizon. Clearly this problem is not trivial because the travel time, and more importantly the acceleration and velocity profiles of a vehicle, are affected by other agents of the traffic network (i.e. traffic lights), whose actions and priorities may conflict with the time constraints of the vehicles.

The analysis is carried out for a simplified urban traffic network with vehicles in free flow, with no constraints coming from other users of the infrastructure. However, we will show that the algorithm may be employed also in more realistic scenarios where the presence of multiple vehicles requires the speed advisories to be updated dynamically.

I2V communication is assumed, and the vehicles have full knowledge of the n traffic lights timings in the driving horizon.

From a mathematical point of view, a constrained optimization problem can be cast. Energy consumption over the trip horizon is to be minimized, subject to the vehicle dynamical model, and time constraints to impose arrivals at the intersections during a green phase.

2.1 Vehicle Model

Vehicles equipped with electric motors will be considered in this work. The equivalent circuit model of a DC motor gives [20]:

$$\begin{aligned} V_a &= i_a R_a + e \\ e &= \kappa \omega_m \\ T_m &= \kappa i_a \end{aligned} \tag{1}$$

where V_a is the armature voltage, i_a is the armature current, R_a is the armature resistance, e is the back electromotive force, κ is the speed constant, ω_m is the rotational speed of the motor, T_m is the electromagnetic motor torque. The motor power supply is converted into mechanical power and electric power loss due to heating of the armature coil, as follows:

$$P = V_a i_a = \omega_m T_m + \frac{R_a}{\kappa^2} T_m^2 \tag{2}$$

where $V_a i_a$ is the electric power supplied to the armature, $\omega_m T_m$ is the actual mechanical power developed by the armature, and $\frac{R_a}{\kappa 2} T_m^2$ is the electric power wasted in the armature. The control input u is represented by the motor torque:

$$u = T_m \tag{3}$$

The vehicle longitudinal dynamical model may be generally written as:

$$m \frac{dv}{dt} = F_t - F_{\text{aero}} - F_{\text{friction}} - F_{\text{slope}} \tag{4}$$

where F_t is the traction force at the wheels, F_{aero} the aerodynamic force, F_{friction} the rolling resistance force, F_{slope} the gravity force.

In the following we assume that there are no losses in the transmission, no slip at the wheels, and that the road slope does not vary in space. In particular, the road slope could be approximated by an average value. Therefore, the vehicle model shall be written as:

$$\begin{cases} \dot{x} = v \\ m\dot{v} = \frac{uR_t}{r} - \frac{1}{2}\rho_a A_f c_d v^2 - mgc_r - mg\sin(\alpha) \end{cases} \tag{5}$$

where R_t is the transmission ratio, r is the wheel radius, m is the vehicle mass, ρ_a is the external air density, A_f is the vehicle frontal area, c_d is the aerodynamic drag coefficient, c_r is the rolling resistance coefficient, α is the road slope, and g is the gravity.

The sum of the aerodynamic and rolling frictions can be approximated as a second order polynomial in the velocity v:

$$F_{\text{res}} = F_{\text{aero}} + F_{\text{friction}} = a_0 + a_1 v + a_2 v^2 \tag{6}$$

where a_0, a_1 and a_2 are parameters identified in previous works [21, 22].

Under these assumptions the vehicle model can be simplified as follows:

$$\begin{cases} \dot{x} = v \\ \dot{v} = h_1 u - h_2 v^2 - h_3 v - h_0 \end{cases} \tag{7}$$

with

$$h_1 = \frac{R_t}{mr}, \quad h_2 = \frac{a_2}{m}, \quad h_3 = \frac{a_1}{m}, \quad h_0 = \frac{a_0}{m} + g\sin(\alpha) \tag{8}$$

Therefore, the power demand can be expressed as:

$$P = b_1 uv + b_2 u^2 \tag{9}$$

where

$$b_1 = \frac{R_t}{r}, \quad b_2 = \frac{R_a}{\kappa^2} \tag{10}$$

2.2 Traffic Lights Timings

In this study the traffic lights are not actuated, therefore cycle time, phase time and offset are deterministic and given. It is possible to formulate mathematically the state evolution of the traffic lights as follows:

$$s_i(\tau) = \begin{cases} 1, & kT < \tau - \theta_i \leqslant kT + T_{\text{gr}} \\ 0, & kT + T_{\text{gr}} < \tau - \theta_i \leqslant (k+1)T \end{cases} \tag{11}$$

where s_i is the state of the i-th traffic light, $k \in \mathbb{Z}$ is the number of cycles, T is the cycle time, T_{gr} is the duration of the green phase, and $\theta_i \in [0, T)$ is the traffic light offset at intersection i, for $i \in \{1, \ldots, n\}$.

2.3 Optimal Control Problem

Finally, the optimization problem may be stated as follows:

$$\begin{cases} \min\limits_{u} & J = \int\limits_{t_0}^{t_f} b_1 uv + b_2 u^2 \, \mathrm{d}t \\ \text{s.t.} & \dot{x} = v \\ & \dot{v} = h_1 u - h_2 v^2 - h_3 v - h_0 \\ & x(t_0) = d_0, \quad x(t_f) = d_{n+1} \\ & x(t_i) = d_i \wedge s_i(t_i) = 1 \\ & v(t_0) = v_0, \quad v(t_f) = v_f \\ & v_{\text{min}} \leq v \leq v_{\text{max}} \\ & u_{\text{min}} \leq u \leq u_{\text{max}} \end{cases} \tag{12}$$

where (t_0, d_0) are the vehicle coordinates at the origin of the driving horizon, (t_f, d_{n+1}) are the coordinates of the destination of the driving horizon, t_i is the crossing time at the i-th intersection, d_i is the position of the i-th intersection, for $i \in \{1, \ldots, n\}$, v_0 and v_f are the desired initial and final velocities.

A solution to a simpler problem, with no traffic lights and a simplified vehicle model, was found analytically in [20]. In the case under analysis, the traffic lights introduce an additional complexity to the problem, expressed mathematically by the constraint:

$$x(t_i) = d_i \wedge s_i(t_i) = 1 \tag{13}$$

Equation (13) imposes a non-stop requirement at the signalized intersection, by specifying that the vehicle has to be at the intersection d_i at time t_i, when the traffic light s_i is green. This constraint may result in disjoint sets due to the presence of multiple available green phases at each intersection, and may affect the convexity of the problem. In other words the constraint (13) defines a non-convex set, and the nonlinear objective function, because of such discontinuity in the constraints, assumes multiple local minima. Therefore, the solution to this optimal control problem has to be sought in a sub-optimal way, by use of algorithms that simplify the control envelope, recover the convexity of the constraints set and solve the optimization advising the driver on the velocity to track.

The problem can be solved in a local optimal way by the DP, provided that the selection of the green phases to be traversed is made a priori. In order to obtain a global optimum, DP should be run extensively on all the possible paths to find the one with minimum cost. However its high computational load is clearly not suitable for online uses. In the following, the results obtained via DP will be used as a benchmark for validation and evaluation of the employed sub-optimal strategy.

3 Optimization Algorithm

The original optimal control problem (12) will be now divided into sub-problems for the simplification of constraint (13) and the formulation of a convex optimization problem. The idea is to identify the best green phase at each intersection, in order to finally optimize over a single path from origin to destination.

The methodology may be summarized as in Fig. 1, and described as follows:

1. Pruning algorithm to identify the set of feasible portions of green phases at each intersection. The notion of feasibility refers to the time intervals that allow not to stop and to drive in compliance with the speed limits. The pruning algorithm will reduce the search space of the best path in the driving horizon.

2. Construction of a weighted directed acyclic graph in the feasibility region to approximate the energy cost of all the possible paths in the driving horizon. Dijkstra's algorithm is then run on the approximating graph in order to identify the "best" path (i.e. most energy efficient). With the selection of a single path in the driving horizon, the optimization problem will become convex.

3. Solution of a simple convex optimization problem to obtain the optimal crossing times at each intersection through the previously selected feasible green phases.

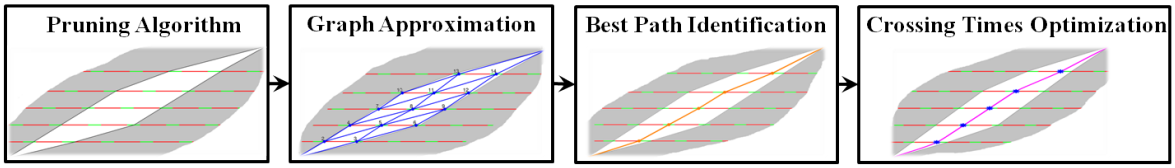

Figure 1: Block diagram representation of the proposed strategy.

3.1 Pruning Algorithm

The velocity pruning algorithm reduces the search space of available green phases by identifying only the feasible ones that allow to reach the destination at the specified final time without stopping, and in compliance with the imposed speed limits.

The algorithm will output $\{t_{i,\text{min}}, t_{i,\text{max}}\}$, for $i \in \{1, \dots, n\}$, which represent respectively the minimum and the maximum feasible crossing times at the i-th intersection. The algorithm may be described as follows:

At each intersection $i \in \{1, \dots, n\}$, the minimum and the maximum feasible crossing times $t_{i,\text{min}}$ and $t_{i,\text{max}}$ are calculated, as indicated in step 2-3 of Algorithm 1. A check on the feasibility of $t_{i,\text{max}}$ is performed in step 4, by checking whether it is possible to reach the destination in compliance with v_{max}. If not feasible, $t_{i,\text{max}}$ is anticipated to the last feasible time instant. The two crossing times $t_{i,\text{min}}$ and $t_{i,\text{max}}$ are verified to belong to a green phase in steps 5-10. If not, they are set to the first instant of the next green phase (line 6) or the last instant of the previous green phase (line 9), respectively. Finally a backward check is performed at each intersection in steps 12-19. If there is any $t_{i,\text{max}} \leq t_{i-1,\text{max}}$ or any $t_{i,\text{max}}$ that induces higher speeds than v_{max}, then $t_{i-1,\text{max}}$ is anticipated to the last feasible time instant (step 14). The new $t_{i-1,\text{max}}$ is verified to belong to a green phase in steps 15-17. If not, it is set to the last instant of the previous green phase.

In case no feasible region is found (i.e. $t_{i,\text{min}} > t_{i,\text{max}}$ for any $i \in \{1, \dots, n\}$), a non-stop trajectory does not exist. However the pruning algorithm is not computationally demanding and may be quickly run again at the beginning of the next green phase or in case of a triggering event (e.g. a sudden stop, an unexpected deceleration, a deviation from the advised velocity, etc.).

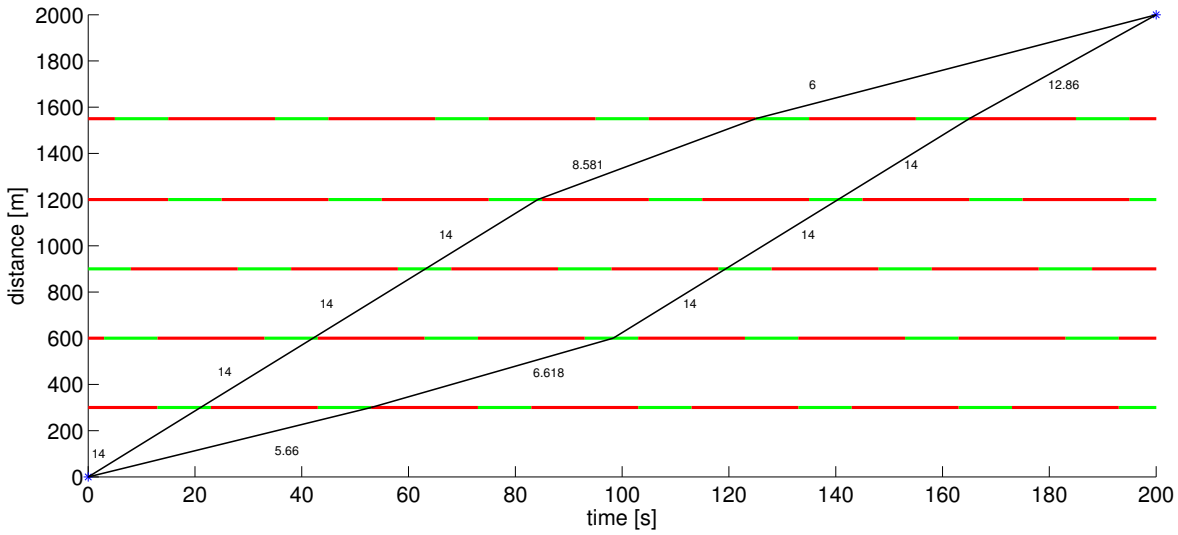

Figure 2: Feasibility region obtained via the pruning algorithm. The labels indicate the speed of the region boundaries, calculated from $t_{i,\min}$ and $t_{i,\max}$. The speed limits were set to $\{v_{\min}, v_{\max}\} = \{5, 14\}$ m/s.

Algorithm 1 Pruning algorithm

for $i \leftarrow 1$ to n **do**
  $t_{i,\min} = \frac{d_i - d_{i-1}}{v_{\max}} + t_{i-1,\min}$;
  $t_{i,\max} = \frac{d_i - d_{i-1}}{v_{\min}} + t_{i-1,\max}$;
  $t_{i,\max} = \min\left\{t_{i,\max}, t_f - \frac{d_{n+1} - d_i}{v_{\max}}\right\}$;
  if $s_i(t_{i,\min}) \neq 1$ **then**
    $t_{i,\min} = \left(\left\lfloor \frac{t_{i,\min}}{T} \right\rfloor + 1\right) T + \theta_i$;
  end if
  if $s_i(t_{i,\max}) \neq 1$ **then**
    $t_{i,\max} = \left\lfloor \frac{t_{i,\max}}{T} \right\rfloor T + T_{\text{gr}} + \theta_i$;
  end if
end for
for $i \leftarrow n$ to 1 **do**
  if $t_{i,\max} \leq t_{i-1,\max}$ || $\frac{d_i - d_{i-1}}{t_{i,\max} - t_{i-1,\max}} > v_{\max}$ **then**
    $t_{i-1,\max} = t_{i,\max} - \frac{d_i - d_{i-1}}{v_{\max}}$;
    if $s_{i-1}(t_{i-1,\max}) \neq 1$ **then**
      $t_{i-1,\max} = \left\lfloor \frac{t_{i-1,\max}}{T} \right\rfloor T + T_{\text{gr}} + \theta_{i-1}$;
    end if
  end if
end for

In Fig. 2 an example of feasible region found by the pruning algorithm is shown. It is easy to observe how the algorithm was capable of reducing the feasible driving domain to the region bounded by the black lines. The algorithm ensures that inside the feasible region there exists a trajectory compliant with the speed limits. All the green phases outside the feasible

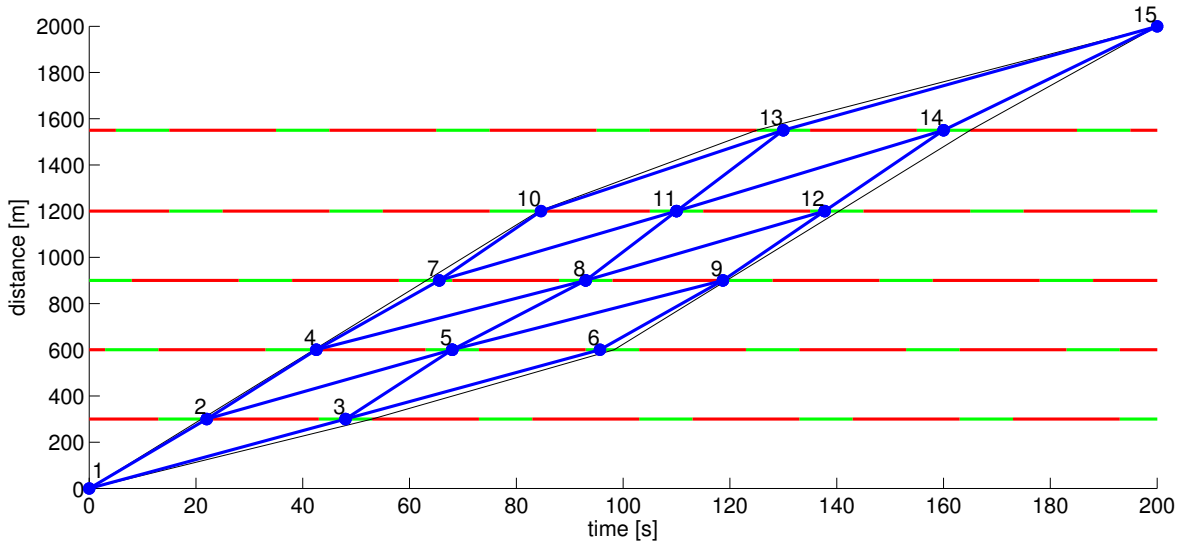

Figure 3: Graph of the possible paths within the feasibility region, with 1 node per green phase (nodes = 15, edges = 22). The labels indicate the ID of the node.

region will be no longer considered because they would induce driving profiles in violation of the speed limits.

3.2 Optimal Path

After running Algorithm 1, the set of feasible green phases is significantly reduced. However many possible trajectories from origin to destination are still present, and constraint (13) still defines disjoint sets, due to the presence of multiple available green phases at each intersection.

The idea is to approximate, by means of a weighted directed acyclic graph, the energy cost of all the possible trajectories from the initial to the final point within the feasibility region, as shown in Fig. 3. The approximation consists in placing the nodes of the graph at specified crossing times in the feasible intervals of the green phases. The number of nodes per feasible green phase depends on the fineness of the approximation. In the case of one node per green phase, the node will be placed at the midpoint of the feasible green interval. In the case of three nodes per green phase, the nodes will be placed at the midpoint and at the two ends of the feasible green interval. In case of a finer approximation, the nodes will be equally spaced, always keeping one node at the phase and two nodes at the two ends of the feasible green interval.

Therefore, let $\mathcal{G} = (N, M)$ be such a graph, where N is the set of vertices (or nodes), and M is the set of feasible edges (or links) connecting the vertices of the graph. Let p_j, for $j = 1, \ldots, |N|$, be the node identifier. Let us also define a weighting function $w : M \to W$, which associates each edge of the graph with a weight. In this application the weight W is the energy cost to travel along the path, and it may be seen as the sum of two contributions:

$$W = E_{\text{total}} = E_{\text{link}} + E_{\text{jump}} \tag{14}$$

The trip on a link is approximated with constant speed and time-invariant power demand.

Therefore, the associated energy consumption is defined as:

$$E_{\text{link}} = \Delta T \left(b_1 \bar{u}_i \bar{v}_i + b_2 \bar{u}_i^2 \right) \tag{15}$$

where $\bar{v}_i = \frac{d_i - d_{i-1}}{\Delta T}$ is the constant velocity on the edge, and $\bar{u}_i = \frac{1}{h_1} \left(h_2 \bar{v}_i^2 + h_3 \bar{v}_i + h_0 \right)$ from (7), for $i \in \{1, \ldots, n+1\}$. The link travel time is defined as $\Delta T = \tau_{p_l} - \tau_{p_j}$, for every edge $(p_j, p_l) \in M$, and $p_j, p_l \in N$, with τ being the crossing time associated with the relative vertex.

The energy consumption associated with the change of velocity at a node between two edges is defined as:

$$E_{\text{jump}} = \int_0^{t_{\text{jump}}} b_1 u(t) v(t) + b_2 u(t)^2 \, \mathrm{d}t \tag{16}$$

where $u(t) = \frac{1}{h_1} \left(\dot{v}(t) + h_2 v(t)^2 + h_3 v(t) + h_0 \right)$ from (7), and $v(t)$ is the time-varying velocity in every transient linearly modeled as $v(t) = \bar{v}_{\text{in}} \pm a \cdot t$, with a being a constant fixed acceleration, $\bar{v}_{\text{in}}$ the constant velocity on the incoming edge to the node. The speed change is assumed to be performed in $t_{\text{jump}} = \frac{|\bar{v}_{\text{out}} - \bar{v}_{\text{in}}|}{a}$, with $\bar{v}_{\text{out}}$ being the constant velocity on the outgoing edge from the node. Note that regenerative braking is not considered in this analysis, therefore E_{jump} is not negative.

The main challenge of such graph approximation is represented by the weight assignment to the edges. Every node of the graph with two or more incoming edges is critical because $\bar{v}_{\text{in}}$ is not unique, and this generates multiple contributions E_{jump} for the outgoing edge. Therefore, the critical nodes need to be "decoupled" in order to have a correct weight assignment.

One option to deal with this problem is to transform the directed acyclic graph into the tree of all the possible paths [19]. This approach has an exponential complexity and its computational load might increase significantly with the size of the approximating graph.

A more efficient solution, in terms of memory allocation and computational load, is proposed in this work, and it is represented by the extension of the original graph into a line digraph [23].

The line graph, $\mathcal{L}(G)$ of a graph $\mathcal{G}$ has as vertices the edges of $\mathcal{G}$, and two vertices of $\mathcal{L}(G)$ are adjacent whenever the corresponding edges of $\mathcal{G}$ are adjacent. The in-degree $\text{id}(p_j)$ of a vertex p_j is the number of edges incident to p_j (the inlines at p_j), while its out-degree $\text{od}(p_j)$ is the number of edges leaving p_j (the outlines at p_j). A point with no inlines is a source, one with no outlines is a sink.

Let $\mathcal{G} = (N, M)$ be the graph with N nodes and M edges as before. Then in $\mathcal{L}(G) = (N^*, M^*)$, the number of nodes is $|N^*| = |M|$ and the number of edges is:

$$|M^*| = \sum_{j=1}^{|N|} \text{id}(p_j) \cdot \text{od}(p_j)$$

The adoption of the line graph allows to solve the criticality of the original graph by intrinsically decoupling the nodes with multiple incoming edges. The problem boils down to assigning to each edge of the line graph a weight corresponding to the sum of E_{link} relative to the edge indicated in the identifier of the destination node, and E_{jump} relative to the speed change when coming from the link indicated in the origin node (see Fig. 4). Furthermore, the size of $\mathcal{L}(G)$ is smaller than the size of the tree of all the possible paths [19]. The number of nodes in the expanded tree grows with an exponential law, whereas the number of nodes of the line graph grows with a polynomial law (namely quadratic).

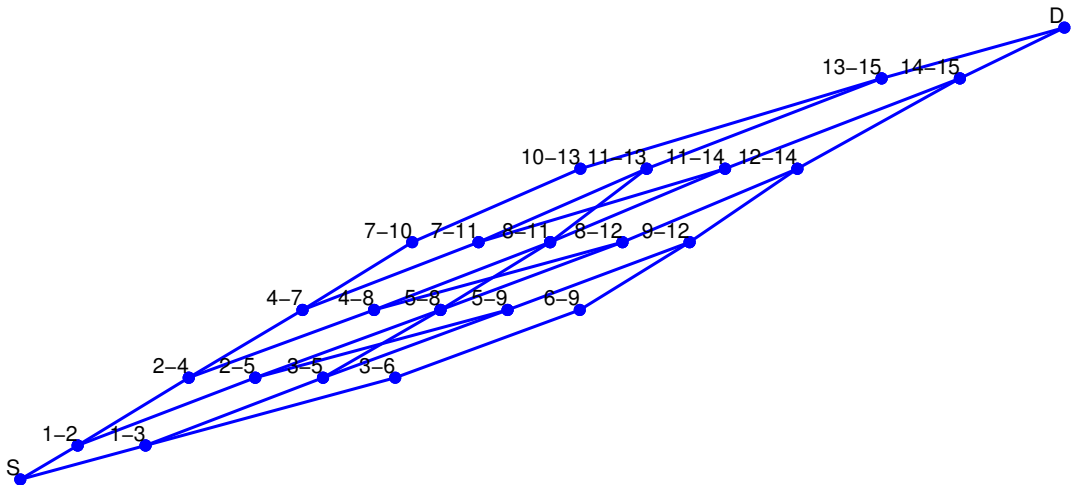

Figure 4: Line graph of the original graph in Fig. 3 (nodes = 24, edges = 34). The labels indicate the node identifier in the line graph, that is the corresponding link in the original graph.

Having as objective the online adoption of the algorithm, further improvements can be achieved by exploiting the specific structure of the graph. The adjacency matrix of a directed acyclic graph is highly sparse, in fact all the diagonal and lower-diagonal terms are zero. The high sparsity of the adjacency matrix leads to very inefficient memory management and matrix operations, therefore it is highly beneficial to define the directed graph as a list of edges.

By adding the two fictitious nodes S and D in the line graph, the structure of a directed graph with one source and one destination is recovered. At this point, Dijkstra's algorithm can be run on the line graph in order to obtain the most energy efficient path from origin to destination on the approximating graph. Dijkstra's algorithm will return the optimal nodes to be traversed, which will be mapped back into the optimal path on the original graph. In Table 1, the possible paths and their identifiers are reported..

Table 1: Possible paths in the original graph

Path	**Path Identifier**	**Path**	**Path Identifier**
1-2-4-7-10-13-15	1	1-2-5-8-12-14-15	9
1-2-4-7-11-13-15	2	1-2-5-9-12-14-15	10
1-2-4-7-11-14-15	3	1-3-5-8-11-13-15	11
1-2-4-8-11-13-15	4	1-3-5-8-11-14-15	12
1-2-4-8-11-14-15	5	1-3-5-8-12-14-15	13
1-2-4-8-12-14-15	6	1-3-5-9-12-14-15	14
1-2-5-8-11-13-15	7	1-3-6-9-12-14-15	15
1-2-5-8-11-14-15	8		

3.3 Simplified Optimization Problem

Once Dijkstra's algorithm has provided the most efficient energy path on the approximating graph, the problem can be finally formulated as a convex optimization. The disjoint sets given by constraint (13) are resolved, and only one time interval per intersection represents now the optimization domain. The mathematical formulation may be carried out in multiple ways. However, defining the objective as a function of speed would result into nonlinear constraints, which would slow down the solution of the problem [19]. A more efficient formalization makes use of the crossing times at each intersection as decision variables of the optimization problem. Thus the constraints become constant.

Let us first define a vector of optimization variables (crossing times at intersections), supposing n to be the number of intersections:

$$\mathbf{x} = [t_1, t_2, \dots, t_n]^T \in \mathbb{R}^n \tag{17}$$

Then a vector of travel times for each segment is defined:

$$\mathbf{t} = \mathbf{t}(\mathbf{x}) = [t_1 - t_0, t_2 - t_1, \dots, t_n - t_{n-1}, t_f - t_n]^T \tag{18}$$

Knowing the position of the traffic lights, let us then define the vector of the constant velocities in each segment:

$$\begin{aligned} \mathbf{v} = \mathbf{v}(\mathbf{x}) &= [v_1, v_2, \dots, v_n, v_{n+1}]^T \\ &= \left[\frac{d_1 - d_0}{t_1 - t_0}, \frac{d_2 - d_1}{t_2 - t_1}, \dots, \frac{d_n - d_{n-1}}{t_n - t_{n-1}}, \frac{d_{n+1} - d_n}{t_f - t_n}\right]^T \end{aligned} \tag{19}$$

where d_i is the position of the i-th traffic light with respect to the origin, d_0 is the origin, d_{n+1} is the destination, t_f is the horizon time, and $\mathbf{t} \in \mathbb{R}^{n+1}$, $\mathbf{v} \in \mathbb{R}^{n+1}$.

The objective function is the same as the one in the original optimization problem (12):

$$J = \int_{t_0}^{t_f} P \, \mathrm{d}t = \int_{t_0}^{t_f} b_1 uv + b_2 u^2 \, \mathrm{d}t \tag{20}$$

By using the same procedure as in the graph weights assignment, we may split the objective function into an energy term related to the segments traveled at constant velocity, and an energy term related to the velocity variations between subsequent segments. Therefore, it may be rewritten as:

$$J(\mathbf{x}) = \mathbf{t}^T \bar{P}(\mathbf{v}) + \sum_{i=0}^{n+1} E_{\text{jump}} \tag{21}$$

where $\bar{P}(\mathbf{V})$ is the power required by the vehicle when traveling at constant velocity (no velocity variation in time):

$$\bar{P}(\mathbf{v}) = b_1 \bar{u}(\mathbf{v})\mathbf{v} + b_2 \bar{u}(\mathbf{v})^2 \tag{22}$$

with

$$\bar{u}(\mathbf{v}) = \frac{1}{h_1}\left(h_2 \mathbf{v}^2 + h_3 \mathbf{v} + h_0\right) \tag{23}$$

Finally the optimization problem may be formulated as:

$$\begin{cases} \min\limits_{\mathbf{x}} \; J(\mathbf{x}) \\ \max\left\{t_i^-, t_{i,\min}\right\} \leq t_i \leq \min\left\{t_i^+, t_{i,\max}\right\} \end{cases} \tag{24}$$

where t_i^- and t_i^+ are constants and represent the end times of the selected green phase at each intersection $i \in \{1, \ldots, n\}$, and $t_{i,\min}$ and $t_{i,\max}$ are the minimum and the maximum feasible crossing times returned by the pruning algorithm.

4 Simulation results

In the following, the performance of the proposed sub-optimal algorithm is compared to the optimal solution provided by DP [24].

Table 2: Simulation parameters

(a) Vehicle parameters

Parameter	Value
m	1190 kg
r	0.2848 m
R_t	6.066
a	1.5 m/s^2
α	0 rad
a_0	113.5 N
a_1	0.774 N/(m/s)
a_2	0.4212 N/(m/s)^2
b_2	0.1515 Ω

(b) Scenario parameters

Parameter	Value
T	30 s
T_{gr}	10 s
n	5
$\theta_{1,\ldots,n}$	[13, 3, 28, 15, 5] s
t_f	200 s
d_{n+1}	2000 m
t_0	0 s
d_0	0 m
$d_{1,\ldots,n}$	[300, 600, 900, 1200, 1550] m
v_f	10 m/s
$v_{\min}$	5 m/s
$v_{\max}$	14 m/s

4.1 Control Scenario

The case under study presents a scenario with only one vehicle in the traffic network. However it can be conceptually extended to the case of multiple vehicles in free flow, each equipped with the presented algorithm. Traffic effects, such as inter-distance constraints and pedestrians cross-walks, are taken care of by new runs of the algorithm.

The vehicle is supposed to be traveling through $n = 5$ signalized intersections, along a road stretch of 2000 m on a total time horizon of 200 s. The distance between the intersections is about 300 m. The choice of the final time and distance of the driving horizon is made in such a way to keep a realistic overall average speed of about $v = 10$ m/s. The vehicle objective is to follow an energy optimal velocity profile which allows to find all the traffic

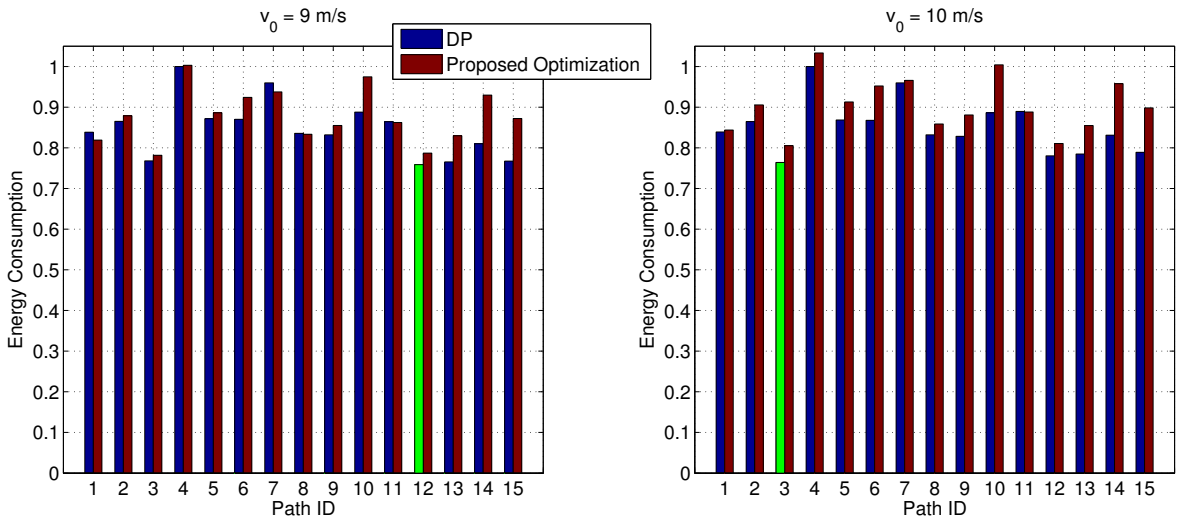

Figure 5: Comparison of DP and proposed strategy energy consumption calculation on all the feasible paths. Two cases with different initial speeds are reported. In green the most energy-efficient path.

lights on green. No assumption is made on the existence of a "green wave". Clearly its existence would enable an easier solution, which the presented algorithm is able to find. The problem is made more challenging and interesting by the random disposition of the green phases, and consequently by the presence of more trip options. The green phase duration is 10 s, the total cycle time is set to 30 s. Note that, although some traffic lights might be phased so to give rise to a "green wave", the distribution of the green phases may be such that the velocity of the wave is not energy optimal.

The initial velocity of the vehicle (i.e. the initial kinetic energy) is varied throughout the simulations to assess its impact on the energy consumption and on the optimal path. Traffic lights timings, final velocity, horizon are fixed for consistency in results comparison. The objective of the performed simulations is to prove the importance of a methodology which allows to achieve a fast sub-optimal solution suitable for online implementation and eco-driving assistance service.

The employed simulation parameters are provided in Table 2.

4.2 Optimal Path Validation

The first set of experiments aims at validating the pruning algorithm and the optimal path identification. DP, used in the following as a benchmark, provides the optimal solution to the problem, and it is used to compute the energy cost of all the possible paths in the feasible region.

Simulations have been run varying the initial velocity of the trip, which has an impact on the optimal path. In the scenario under study the vehicle does not stop at any intersection in the feasibility region shown in Fig. 3.

In Fig. 5 the energy consumption for each path computed via DP is compared with the energy consumption approximation yielded by the proposed strategy. The results for two different choices of initial velocity are reported after normalization with respect to the maximum DP-cost path in each configuration. The minimum cost path, for each case, is

Table 3: Optimal path identification (Graph vs. DP)

Graph \ v_0 [m/s]	5	6	7	8	9	10	11	12	13	14
1 node/green phase	**15**	**15**	**15**	**15**	**3**	**3**	**3**	**3**	**3**	**3**
3 nodes/green phase	**12**	**12**	**12**	**12**	**12**	**3**	**3**	**3**	**3**	**3**
DP (optimal path)	**12**	**12**	**12**	**12**	**12**	**3**	**3**	**3**	**3**	**3**

highlighted in green. The path ID in abscissa refers to Table 1.

The energy cost approximation obtained with the proposed strategy is very close to the true energy cost computed with DP. The root-mean-square error (RMSE) of our approximation with respect to the true cost was calculated for the two experiments shown in Fig. 5. It has been then normalized with respect to the mean value of the energy costs calculated via DP for each case. In the case of initial velocity $v_0 = 9$ m/s, the normalized RMSE is 6.3%. In the case of $v_0 = 10$ m/s, the normalized RMSE is 7.7%.

The energy consumption varies depending on the path choice; in the worst case, the choice of a path rather than the optimal one, results to be up to 25% more expensive in terms of energy. In Fig. 5 the DP and the proposed strategy results for only $v_0 = 9$ m/s and $v_0 = 10$ m/s are reported. For speeds $v_0 \geq 10$ m/s, the energy-optimal path is always path number 3; for speeds $v_0 \leq 9$ m/s the optimal path is always path number 12.

The accuracy of the graph approximation, as well as the computational load, increases with the number of nodes per green phase. A graph with one node per green phase, as shown in Fig. 3, and a graph with three nodes per green phase (one at the mid-point and two at the ends) have been tested, and the optimal path identification obtained with Dijkstra's algorithm is validated against the true minimum-energy path, found via DP.

As summarized in Table 3, the graph with 3 nodes per green phase approximates better the whole energy cost, never failing, whereas the smaller graph fails in five cases. Clearly there exist other graph configurations where the approximation with 3 nodes per green phase does not always provide the right identification of the energy-optimal path. One can increase the discretization level (i.e. use more nodes per green phase), increasing consequentially also the computational load. Thanks to the newly introduced line graph expansion, the computational time does not grow as dramatically as with the tree expansion, so this becomes more affordable. Nevertheless this is a sub-optimal solution, the path identification is not flawless, and a trade-off between fineness of discretization and computational effort is required. However, it is possible to note that even a graph with a less fine discretization is able to provide a path identification that is not far from the true optimal one. For instance, as reported in Table 3, the graph with 1 node per green phase fails for $v_0 = 9$ m/s, suggesting path number 3, which is only 2% more costly than the true optimal one, as shown in Fig. 5.

4.3 Optimal Crossing Times

Finally, once Dijkstra's algorithm provides the optimal green phases, the optimal crossing times at each intersection are to be calculated, in order to provide the eco-driving assistance by suggesting a velocity profile to the driver. The solver employed to solve the optimization problem (24) in MATLAB® is `fmincon`, using the interior-point method algorithm.

In Fig. 6, the optimal solution provided by DP through the optimal green phases (path 12

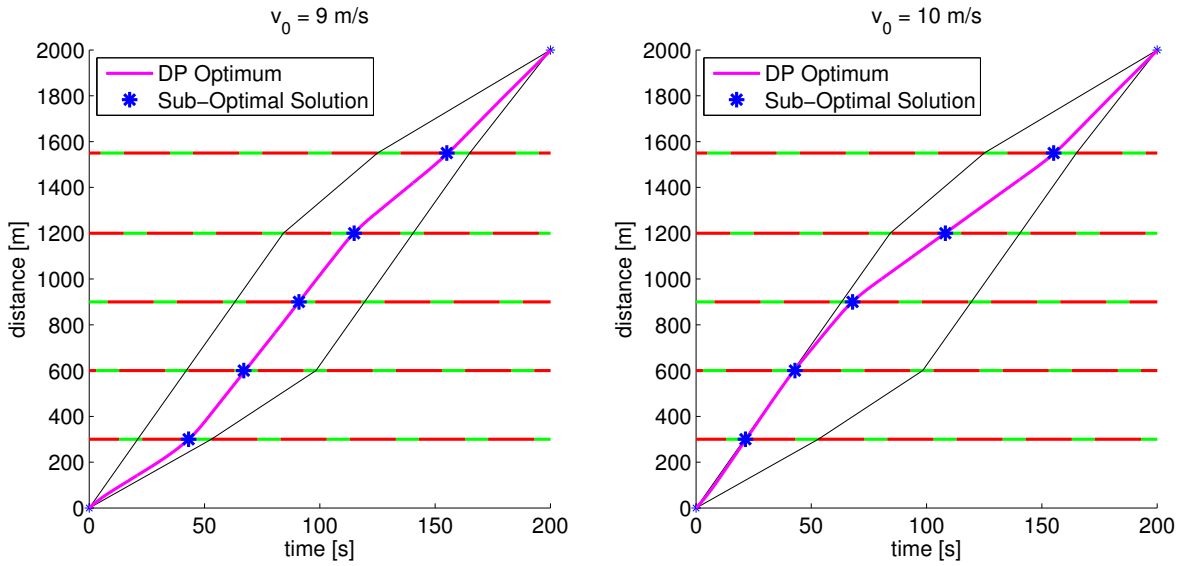

Figure 6: Sub-optimal crossing times compared to the optimal solution provided by DP, for two different initial speeds.

Table 4: Time deviation from the optimal crossing times

	Int. #1	**Int. #2**	**Int. #3**	**Int. #4**	**Int. #5**
$v_0 = 9$ m/s	0.03 s	0.62 s	0.03 s	0.07 s	0.5 s
$v_0 = 10$ m/s	0.22 s	0.36 s	0.05 s	0.08 s	0.4 s
Overall average deviation: 0.23 s					

for $v_0 = 9$ m/s and path 3 for $v_0 = 10$ m/s) is compared to the sub-optimal proposed solution. In Table 4 the time deviations of the sub-optimal solution from the DP solution at the five intersections are summarized, for the two different initial velocities.

The simulations were run with a laptop equipped with an Intel(R) Core(TM) i7-2760QM at 2.40GHz and 8GB of RAM. DP takes about 2.5 hours per path to provide the optimal solution. In order to have an assessment of the cost of all paths and find the minimum-energy one, DP has to be run as many times as the number of possible paths. The proposed methodology takes 1.2 seconds (graph 3 nodes/green phase) or 0.85 seconds (graph 1 node/green phase) to run all the steps presented in Fig. 1 and provide the sub-optimal crossing times at each intersection.

4.4 Robustness of the Algorithm

As already mentioned, the algorithm is capable of coping with unexpected deviations from the suggested trajectory and handling new information coming from the traffic network in a receding-horizon fashion. The horizon can be thought of as the range of the wireless I2V communication.

The following simulation results show how new runs of the algorithm can provide new speed advisory in case of triggering events.

Let us recall once again, on the left-hand side of Fig. 7, Dijkstra's optimal path relative

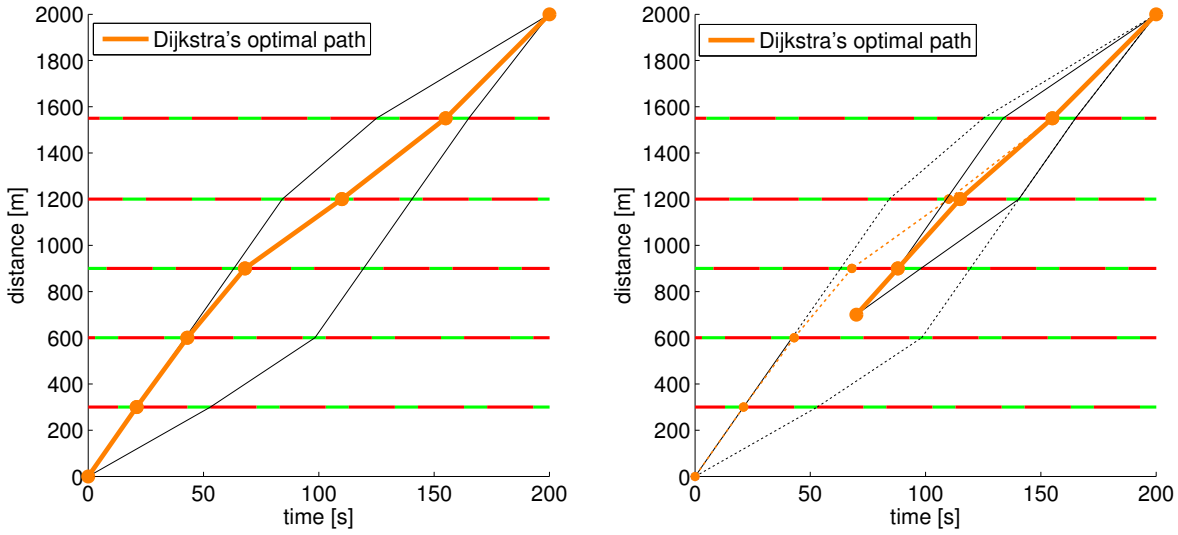

Figure 7: The graph on the left shows the path suggested by Dijkstra's algorithm for $v_0 = 10$ m/s, when going from the origin $(0, 0)$ to the horizon $(200, 2000)$. The graph on the right shows the suggested path when the origin is moved to the point of coordinates $(70, 700)$.

to the simulation scenario described so far, and for an initial velocity $v_0 = 10$ m/s. Let us suppose now that an unexpected event occurs at time $t = 70$ seconds, and the equipped vehicle has to slow down or stop, thus deviating from the suggested trajectory. A new run of the algorithm shows that it is still possible to recover from the perturbation and reach the desired horizon at the desired final time. Evidently, the new feasibility region is smaller, the approximating graph also will be smaller, leading to a smaller set of available paths and a faster solution of the algorithm. On the right-hand side of Fig. 7, the new Dijkstra's optimal path is shown, while the old one is left dotted in the background for the sake of comparison.

Let us show now an example of the receding-horizon capabilities of the algorithm. The coordinates of the horizon are assumed to change as new information from the traffic network is collected. Let the point of coordinates $(70, 700)$ be the new origin of the driving horizon, that is the point of the occurred perturbation as discussed above. New timings information are received from the traffic lights and two new intersections appear in the driving horizon, which now has as destination the point of coordinates $(230, 2500)$. The algorithm is run on the new horizon, defining a new feasibility region, a new approximating graph, and providing again the optimal crossing times at each intersection in order to reach the new destination without stopping. On the left-hand side of Fig. 8, Dijkstra's optimal path is shown, in case the vehicle is able to recover quickly from the perturbation and be at the new origin with an initial velocity $v_0 \geq 9$ m/s. However, if the vehicles is forced to stop or slow down considerably ($v_0 < 9$ m/s), the optimal trajectory to reach the desired destination at the desired time is reported on the right-hand side of Fig. 8.

4.5 Microscopic Traffic Simulator

As mentioned, in this work the energy-optimal trajectory is obtained under the assumption of one vehicle in free flow conditions (i.e. unperturbed by the presence of other vehicles). The assumption was made for the purpose of a simpler and more abstract analysis of the optimization problem. However, the strategy robustness properties demonstrated that the

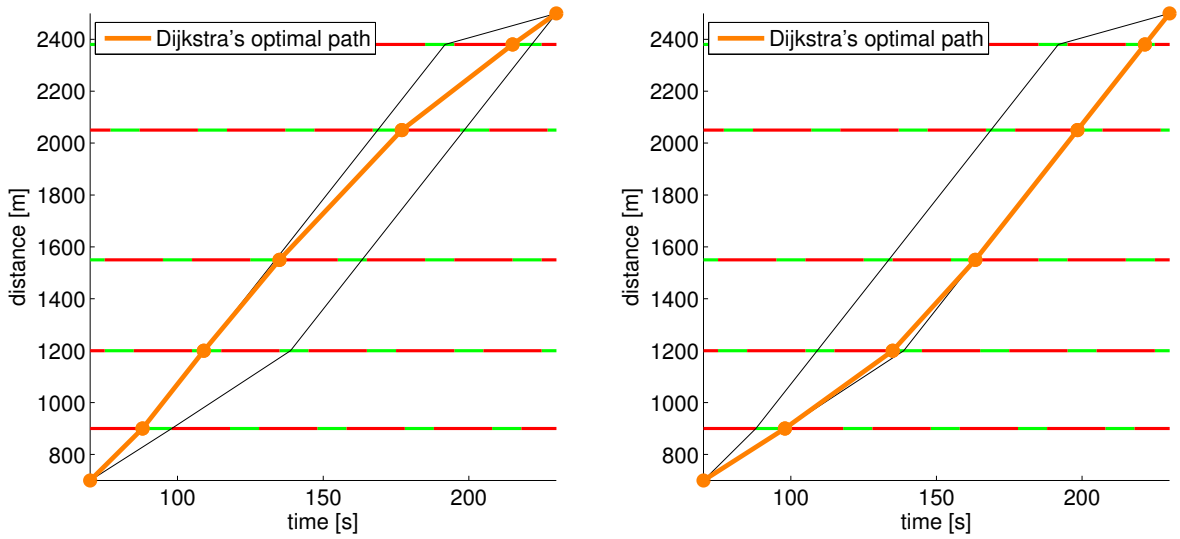

Figure 8: The graph on the left shows the path suggested by Dijkstra's algorithm for $v_0 \geq 9$ m/s, when going from the new origin $(70, 700)$ to the new horizon $(230, 2500)$. The graph on the right shows the suggested path when the initial velocity is $v_0 < 9$ m/s.

algorithm is capable of dealing with perturbations coming from other agents of the traffic network. The optimal path is quickly recalculated in the event of deviation from the original speed advisory.

In order to prove the actual capabilities of the proposed strategy in a more realistic scenario, the traffic microscopic simulator Aimsun has been used. The road topology treated in this work and shown in Fig. 2 has been faithfully replicated in Aimsun, as well as the traffic lights position and signal timings. A flow of 400 vehicles per hour has been set to enter the road network at the initial speed $v_0 = 10$ m/s. The vehicles have been equipped with the algorithm here presented and, evidently, the assumption of unperturbed vehicles in such realistic scenario does not hold anymore. In particular, being each vehicle independently equipped with the algorithm and entering the network at different time instants, the speed advisories may conflict with one another causing deviations from the optimal speed advisory.

The experiment has been set up in such a way that only one lane is available and no pass is allowed, in order to generate as many perturbations as possible to the speed advisories. Note that the space coordinate of the origin d_0 is the same for all vehicles, whereas the time coordinate of the origin t_0 is different for every vehicle. This evidently affects the time coordinate of the destination of the driving horizon, but not its time span, which is the same for all vehicles. The online adaptation of the proposed strategy in the multi-vehicle scenario is quite simple and intuitive. When a new vehicle enters the network, the algorithm is run and the optimal crossing times at each intersection are calculated. The optimal crossing times are then converted in speed advisory, which is continuously updated according to the current position of the vehicle. When the suggested speed grows larger than the maximum speed limit, meaning that a perturbation has occurred, the algorithm is run again. The time horizon is not extended if a new optimal trajectory can be found, otherwise small successive extensions of the time horizon are recursively tried until an optimum is found.

Several experiments were conducted at different levels of penetration rate of the proposed technology (i.e. percentage of equipped vehicles), in order to assess the impact of the proposed algorithm on the traffic energy consumption. For the sake of results consistency

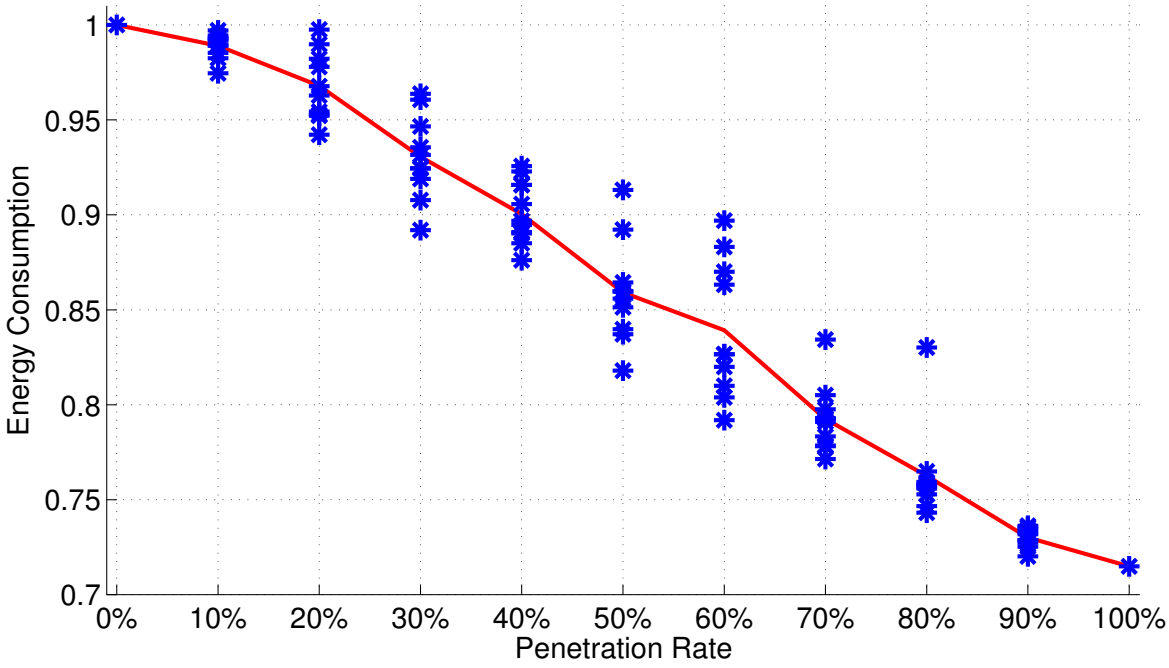

Figure 9: Variation of traffic energy consumption with respect to the technology penetration rate. The red line describes the average energy consumption at the different penetration rates.

Table 5: Traffic performance metrics

	Energy Consumption	Travel Time	N. of Stops	Idling Time
Penetration Rate 0%	2*10E8 J	213.9 s	2.7	37.9 s
Penetration Rate 100%	1.43*10E8 J	205.7 s	0	0 s

and comparison, the vehicle average travel time is set to be the same independently of the penetration rate. In other words, the proposed algorithm was set up in order to achieve the same travel time as in the uncontrolled case (i.e. all vehicles traveling at the maximum speed limit). A statistical analysis conducted for the different levels of technology penetration rate demonstrated that the average travel time can be considered as constant: mean value of 208.5 seconds, and standard deviation of 3.8 seconds (i.e. 1.8% of the mean value).

In Fig. 9, the variation of traffic energy consumption with respect to the technology penetration rate is shown. The proposed strategy has been applied to a gradually increasing number of vehicles in the road network. At penetration rate equal to 0%, the experiment has been conducted with no vehicle equipped with the algorithm, and all the vehicles were allowed to drive up to the maximum speed limit. For penetration rates greater than 0%, a total of 10 simulations per each tested value has been run. The equipped vehicles were chosen randomly, and it is possible to observe that the order of "smart" vehicles in traffic can affect significantly the energy consumption. As expected, a penetration rate of 100% results to be highly beneficial with respect to the uncontrolled case in terms of several traffic performance metrics (see Table 5). A reduction of 28.5% in traffic energy consumption has been observed, without affecting the average travel time. More specifically, the average travel time results to be even smaller than the uncontrolled case by 4%. The stops and the idling time are completely eliminated.

One may argue that a global deployment of the proposed strategy would be difficult to achieve in a close future. A penetration rate of 40% is more realistic and results to be already appealing in terms of energy consumption, yielding an average reduction of about 10%.

5 Conclusions

The exchange of information between the infrastructure and the vehicles (I2V) has been proved in literature to be beneficial in terms of traffic energy consumption. This work focuses on the possibility of further improvements when information about many successive signalized intersections is available. The presented algorithm is capable of finding the energy-efficient path among all the available ones, and returning the speed advisory to the drivers, in a sub-optimal way. The length of the optimization horizon can be arbitrarily long. The only limitations come from the I2V communication range and/or online execution constraints. However, for the already complex scenario analyzed in this study, the computation time required by the algorithm is very appealing for online implementation purposes. The robustness capabilities of the algorithm have been also presented, by showing how perturbations and deviations from the suggested trajectory are handled. In particular, experiments in a microscopic traffic simulator demonstrated that, even in presence of multiple vehicles, the proposed algorithms is able to cope with perturbations and to drastically reduce the traffic energy consumption without affecting travel time.

The experiments in the microscopic traffic simulator highlighted the fact that a performance improvement is achievable even though the vehicles are independently equipped with the proposed algorithm and do not exchange information among them. Further improvements could be achieved by exploiting V2V communication and by sharing information among the vehicles about their own optimal trajectory. Moreover, if the traffic flow grows larger, congestion and queues become inevitable. More sophisticated methods would be required, such as macroscopic models for the estimation of the queue length. In such a situation the problem of eco-driving changes its nature, and a different analysis would be necessary. Lastly, an energy consumption model for electric vehicles was considered in this work. Different energy and/or emissions models could be introduced into the problem, without any loss of generality for the proposed strategy.

References

[1] What Is the EU Doing about Climate Change? URL http://ec.europa.eu/clima/policies/brief/eu/.

[2] Schrank D, Eisele B, Lomax T. TTI's 2012 Urban Mobility Report. *Technical Report*, Texas A\&M Transportation Institute 2012.

[3] Goel A, Kumar P. Characterisation of Nanoparticle Emissions and Exposure at Traffic Intersections through Fast–Response Mobile and Sequential Measurements. *Atmospheric Environment* 2015; .

[4] Hunt PB, Robertson DI, Bretherton RD, Winton RI. SCOOT - A Traffic Responsive Method of Co-Ordinating Signals. *Technical Report*, TRRL Laboratory Report 1014 1981.

[5] Diakaki C. Integrated Control of Traffic Flow in Corridor Networks. PhD Thesis, Technical University of Crete 1999.

[6] Hartenstein H, Laberteaux KP. A Tutorial Survey on Vehicular Ad Hoc Networks. *IEEE Communications Magazine* 2008; .

[7] Trayford RS, Doughty BW, Wooldridge MJ. Fuel Saving and Other Benefits of Dynamic Advisory Speeds on a Multi-Lane Arterial Road. *Transportation Research Part A* 1984; **18**(5-6).

[8] Trayford RS, Doughty BW, van der Touw JW. Fuel Economy Investigation of Dynamic Advisory Speeds from an Experiment in Arterial Traffic. *Transportation Research Part A* 1984; **18**.

[9] Katsaros K, Kernchen R, Dianati M, Rieck D. Performance Study of a Green Light Optimized Speed Advisory (GLOSA) Application Using an Integrated Cooperative ITS Simulation Platform. *International Wireless Communications and Mobile Computing Conference*, 2011.

[10] Li M, Boriboonsomsin K, Wu G, Zhang WB, Barth M. Traffic Energy and Emission Reductions at Signalized Intersections: A Study of the Benefits of Advanced Driver Information. *International Journal of ITS Research* 2009; **7**(1):49–58.

[11] Mandava S, Boriboonsomsin K, Barth M. Arterial Velocity Planning Based on Traffic Signal Information under Light Traffic Conditions. *IEEE Conference on Intelligent Transportation Systems*, 2009.

[12] Mahler G, Vahidi A. Reducing Idling at Red Lights Based on Probabilistic Prediction of Traffic Signal Timings. *IEEE American Control Conference*, 2012.

[13] Koukoumidis E, Martonosi M, Peh LS. Leveraging Smartphone Cameras for Collaborative Road Advisories. *IEEE Transactions on mobile computing* 2012; **11**(5):707–723.

[14] Asadi B, Vahidi A. Predictive Cruise Control: Utilizing Upcoming Traffic Signal Information for Improving Fuel Economy and Reducing Trip Time. *IEEE Transactions on Control Systems Technology* 2011; **19**(3):707–714.

[15] Ozatay E, Ozguner U, Filev D, Michelini J. Analytical and Numerical Solutions for Energy Minimization of Road Vehicles with the Existence of Multiple Traffic Lights. *IEEE Conference on Decision and Control*, 2013.

[16] Rakha H, Kamalanathsharma RK. Eco-Driving at Signalized Intersections Using V2I Communication. *IEEE Conference on Intelligent Transportation Systems*, 2011.

[17] Miyatake M, Kuriyama M, Takeda Y. Theoretical Study on Eco-Driving Technique for an Electric Vehicle Considering Traffic Signals. *IEEE Conference on Power Electronics and Drive Systems*, 2011.

[18] Seredynski M, Mazurczyk W, Khadraoui D. Multi-Segment Green Light Optimal Speed Advisory. *IEEE 27th International Symposium on Parallel and Distributed Processing Workshops and PhD Forum*, 2013.

[19] De Nunzio G, Canudas De Wit C, Moulin P, Di Domenico D. Eco-Driving in Urban Traffic Networks Using Traffic Signal Information. *IEEE Conference on Decision and Control*, 2013.

[20] Petit N, Sciarretta A. Optimal Drive of Electric Vehicles Using an Inversion-Based Trajectory Generation Approach. *IFAC World Congress*, 2011.

[21] Dib W, Chasse A, Moulin P, Sciarretta A, Corde G. Optimal Energy Management for an Electric Vehicle in Eco-Driving Applications. *Control Engineering Practice* 2014; **29**:299–307.

[22] Dib W, Chasse A, Di Domenico D, Moulin P, Sciarretta A. Evaluation of the Energy Efficiency of a Fleet of Electric Vehicle for Eco-Driving Application. *Oil & Gas Science and Technology – Rev. IFP Energies nouvelles* 2012; **67**(4):589–599.

[23] Harary F, Norman RZ. Some Properties of Line Digraphs. *Rendiconti del Circolo Matematico di Palermo* 1960; **9**(2):161–168.

[24] Sundström O, Guzzella L. A Generic Dynamic Programming Matlab Function. *IEEE International Conference on Control Applications*, 2009.