A graph-based algorithm for optimal control of switched systems: An application to car parking

Mattia Laurini, Luca Consolini and Marco Locatelli

***Abstract*—We consider a finite element approximation of the Bellman equation for the optimal control of switched systems. We show that the problem belongs to a special class that we studied in a previous work, for which we developed an efficient solution algorithm. As an application, we present the problem of generating parking maneuvers for self-driving vehicles on two typical urban parking scenarios. The vehicle is described by four different switched systems in which every switching is associated to a penalization term. In this way, we obtain parking paths that have a small number of direction changes and have a simple structure.**

I. Introduction

As a motivating example, consider the following kinematic car-like model with rear-wheel drive and front steering: $(\dot{w},\dot{y},\dot{\theta}) = (v\cos\theta, v\sin\theta, \omega)$, where (w,y) is the position of the center of the real wheel axle, θ the orientation angle, and pair (v,ω) the control input, where v and ω are the linear and angular velocity, respectively.

We aim at finding a minimum-length maneuver that drives the car from an initial configuration to a final parking position, in presence of obstacles.

In literature, there are various approaches for solving this optimal planning problem (see, for instance, [1], [2], [3], [4]). Some of them are based on geometric observations on specific parking scenarios, like [5], [6], which introduce algorithms for a parallel parking lot, or [7] for a perpendicular one. Others address the problem by means of interpolating curves, like [8], [9], [10]. Some works adopt heuristic strategies, like [11], [12], probabilistic approaches, like [13], or randomized techniques, like [14], [15], [16], in which extensions of the Rapidly-exploring Random Tree (RRT) algorithm are proposed, whilst others exploit the Pontryagin's Maximum Principle, like [17]. For a more comprehensive overview on motion planning techniques see, for instance, [18].

In this work, we decide to follow a different approach which consists on the numerical solution of the Bellman equation associated to the optimal planning problem (see, for instance, [19], [20]). With respect to geometric methods tailored to specific parking configurations, such as [6], [7], this approach has two main advantages. First, the method is very general and the solution procedure is independent on the initial configuration of the vehicle and the specific parking scenario. Second, the solution of the Bellman equation allows to compute the optimal paths starting from *all* initial conditions and allows to easily define a feedback control policy. Moreover, unlike probabilistic or heuristic approaches (such as RRT algorithm), this is deterministic and guarantees the optimality of the obtained solution. The main disadvantage of this approach is the solution time. In fact, as we will see in Section III, it requires the solution of an equation with a large number of unknowns. Nonetheless, as shown in Section VI, computational times are getting closer to real-time requirements and are already suited for an automated-parking system. Further improvements could be done both on software and hardware acceleration.

All authors are with the Dipartimento di Ingegneria e Architettura, Università degli Studi di Parma, Parco Area delle Scienze 181/A, 43124 Parma, Italy. E-mails: {mattia.laurini, luca.consolini, marco.locatelli}@unipr.it

The numerical solution of the Bellman equation may lead to parking trajectories that exhibit too many changes from forward to backward direction due to discretization error. One possible way to solve this problem is to associate a penalty to each switching from forward to backward direction and vice versa. Mathematically, this corresponds to describe the vehicle by a switched system that commutes between the two subsystems:

$$\begin{aligned} &\text{F: } (\dot{w},\dot{y},\dot{\theta}) = (v\cos\theta, v\sin\theta, \omega)\,,\\ &\text{B: } (\dot{w},\dot{y},\dot{\theta}) = (-v\cos\theta, -v\sin\theta, \omega)\,. \end{aligned}$$

The first subsystem (F) corresponds to forward direction and the second one (B) to backward direction. The controller is able to switch from one subsystem to the other at arbitrary times. We associate this switched system to a finite state machine with two states, corresponding to the forward and backward subsystems. We associate a penalty to each transition; this allows keeping the number of direction changes limited. We want to minimize the cost function $C + K\bar{\ell}$, where C is the path length, K the number of direction switchings and $\bar{\ell}$ the penalty term associated to each switching.

More generally, a switched control system consists of a set of controlled subsystems associated to the states of a finite state machine. This framework allows modeling various control systems arising in different applications ([21], [22], [23], [24], [25]). Among works addressing optimal control of switched systems, like [26], [27], [28], [29], [30], in this paper we follow the dynamic programming approach presented in [31], [32], based on the solution of the associated HJB equation. In particular, we use a finite dimensional approximation of space state and admissible controls (similar to the procedures used in [33], [34]). In this way, we find a finite dimensional approximation of the Bellman equation which belongs to a class of problems we studied in [35] and for which we presented an efficient solution algorithm.

Statement of contribution. We show that it is possible to derive a formulation of a discretized version of Bellman equation associated to the solution of optimal control problems of switched systems that falls into a more general class of optimization problems we studied in [36]. This reformulation allows to exploit the iterative algorithm introduced in that work for efficiently solving the problem.

We present an application to the generation of parking maneuvers for self-driving vehicles. We model the vehicle as a switched system in which each switching represents a change to a different type of motion. In this way, for instance, we are able to obtain maneuvers with a limited number of direction changes.

Here, we generalize [37] by introducing the finite state machine approach and by placing the specific application of the parking maneuver into a more general framework. We also extend [38] by including all technical details and proofs and by using four different models for the finite state machines associated to the vehicle.

Notation. Denote by $\mathbb{R}_+ := [0,+\infty)$ the set of nonnegative real numbers. Given $x \in \mathbb{R}^N$ and $A \in \mathbb{R}^{N \times M}$, denote the transpose of x with x^T, for $i \in \{1,\dots,N\}$ the i-th component of x with $[x]_i$, for $j \in \{1,\dots,M\}$ denote the (i,j)-th element of A with $[A]_{i,j}$.

$\|\cdot\|$ represents both the sup norm for real-valued functions and the ∞-norm on $\mathbb{R}^N$, unless explicitly specified. Given a finite set S, the cardinality of S is denoted by $|S|$ and symbol $\varnothing$ denotes the empty set.

Finally, given an alphabet Σ, we denote the set of all strings of length $i \in \mathbb{N}$ over Σ as Σ^i, and the Kleene closure of Σ as Σ^*. Following [39], [40], define a finite state machine as a tuple $\mathbb{A} = (\mathbb{I}, \Sigma, \rho, \iota_0, \mathbb{I}_F)$ where $\mathbb{I} \neq \varnothing$ is a finite set of states, $\Sigma \neq \varnothing$ is a finite set of symbols representing an alphabet, $\iota_0 \in \mathbb{I}$ is the initial state, $\mathbb{I}_F \subseteq \mathbb{I}$ is the set of accept states and $\rho : \mathbb{I} \times \Sigma^* \to \mathbb{I}$ is the transition function acting as follows: let symbol ε denote the empty string, then $(\forall \iota \in \mathbb{I})\ \rho(\iota,\varepsilon) = \iota$, $(\forall \iota \in \mathbb{I})\ (\forall \sigma \in \Sigma)\ \rho(\iota,\sigma) \in \mathbb{I}$ and, let $\tau\sigma$ being the concatenation of string τ and symbol σ, then $(\forall \iota \in \mathbb{I})\ (\forall \sigma \in \Sigma)\ (\forall \tau \in \Sigma^*)\ \rho(\iota,\tau\sigma) = \rho(\rho(\iota,\sigma),\tau)$. A language over Σ is a subset of Σ^* and the language accepted by $\mathbb{A}$ is defined as follows: $L(\mathbb{A}) = \{s \in \Sigma^* \mid (\forall \iota \in \mathbb{I})\ \rho(\iota,s) \in \mathbb{I}_F\}$.

II. Problem Formulation

Definition II.1. *A switched control system is a finite state machine $\mathbb{A} = (\mathbb{I}, \Sigma, \rho, \iota_0, \mathbb{I}_F)$, together with difference equation*

$$\begin{aligned} x(k+1) &= f(x(k),\iota(k),u(k)) \\ \iota(k+1) &= \rho(\iota(k),\sigma(k)), \end{aligned} \tag{1}$$

where $f : X \times \mathbb{I} \times U \to X$ represents the system dynamics, $X \subset \mathbb{R}^n$ is compact, $x = x(k) \in X$ is the system state, $\iota = \iota(k) \in \mathbb{I}$ is the discrete state, $U \subset \mathbb{R}^m$ is a compact set of admissible controls, $u = u(k) \in U$ is the control input and $\sigma(k) \in \Sigma \cup \{\varepsilon\}$ is the input symbol to the finite state machine at time step k, where ε denotes the empty string.

Define: the extended state space $\Xi := X \times \mathbb{I}$; the extended state $\xi : \mathbb{N} \to \Xi$, such that $\xi(k) := (x(k),\iota(k))$ associates to each time step k a system state $x(k)$ and a discrete state $\iota(k)$; the extended input set $\Upsilon := U \times (\Sigma \cup \{\varepsilon\})$; and the extended control input $\upsilon : \mathbb{N} \to \Upsilon$, such that $\upsilon(k) := (u(k),\sigma(k))$ associates to each time step k a control $u(k) \in U$ and a symbol $\sigma(k) \in \Sigma \cup \{\varepsilon\}$. Then, system (1) can be rewritten in form

$$\xi(k+1) = \phi(\xi(k),\upsilon(k)), \tag{2}$$

where $\phi : \Xi \times \Upsilon \to \Xi$ is defined as follows: let $\xi := (x,\iota) \in \Xi$ and $\upsilon := (u,\sigma) \in \Upsilon$, then $\phi(\xi,\upsilon) := (f(x,\iota,u),\rho(\iota,\sigma))$.

Consider the following cost function

$$J(\xi_0,\upsilon) = \sum_{k \in \mathbb{N}} \beta^k \ell(\xi(k),\upsilon(k)), \tag{3}$$

where $\beta \in (0,1)$ is a discount factor and $\ell : \Xi \times \Upsilon \to \mathbb{R}_+$ is a cost functional. For any $\xi = (x,\iota) \in \Xi$ and $\upsilon = (u,\sigma) \in \Upsilon$, ℓ, for instance, may have the form $\ell(\xi,\upsilon) = \ell_1(x,u) + \ell_2(\iota,\sigma)$, where $\ell_1 : X \times U \to \mathbb{R}_+$ is the cost associated to the system and $\ell_2 : \mathbb{I} \times (\Sigma \cup \{\varepsilon\}) \to \mathbb{R}_+$ is the cost associated to the finite state machine transitions.

Assumption II.2. *Functions f and ℓ are differentiable with respect to their continuous variables.*

For any $L \in \mathbb{R}^+$, let $\mathscr{V}_L \subset \{V \mid V : \Xi \to \mathbb{R}\}$ denote the set of real valued functions on Ξ that are L-Lipschitz on X, that is, $(\forall V \in \mathscr{V}_L)\ (\forall x_1,x_2 \in X)\ (\forall \iota \in \mathbb{I})\ |V((x_1,\iota)) - V((x_2,\iota))| \leq L\|x_1 - x_2\|$, and define the Bellman operator $T : \mathscr{V}_L \to \mathscr{V}_L$ as follows: $(\forall V \in \mathscr{V}_L)\ (\forall \xi \in \Xi)$

$$T[V](\xi) := \min_{\upsilon \in \Upsilon} \{\beta V(\phi(\xi,\upsilon)) + \ell(\xi,\upsilon)\}\,. \tag{4}$$

It is well known (see, for instance, [41]) that the function that associates to $\upsilon \in \Upsilon$ the minimum with respect to υ of $J(\xi,\upsilon)$ defined as in (3) corresponds to the fixed point of Bellman operator T defined in (4).

In order to numerically find a fixed point of (4), we discretize sets X and U as follows. Let $\bar{X} := \{x_1,\dots,x_N\} \subset X$, with $N \in \mathbb{N}$, be a set of points that defines a triangulation of X. Let δ_x be the maximum edge length of the triangulation. Similarly, set $\bar{U} := \{u_1,\dots,u_M\} \subset U$, with $M \in \mathbb{N}$, and let δ_u be the maximum of the minimum distances between each element of $\bar{U}$ and the remaining ones. Set $\bar{\Xi} := \bar{X} \times \mathbb{I}$, and set $\bar{V} : \bar{\Xi} \to \mathbb{R}$. In the following, $\bar{V}(\xi)$ will approximate $V(\xi)$ for any $\xi \in \bar{\Xi}$. For $\iota \in \mathbb{I}$, $x \in X$, set $\xi := (x,\iota)$ and define a linearly interpolated cost function $\Lambda_{\bar{V}}(\xi)$ as follows. Let $\{p_1,\dots,p_S\} \subset \bar{X}$, with $S \in \mathbb{N}$, be the extremal points of the simplex of the triangulation containing x. Then, x can be uniquely written as $x = \sum_{j=1}^{S} \alpha_j p_j$, with $\alpha_j \in [0,1]$, such that $\sum_{j=1}^{S} \alpha_j = 1$, and set $\Lambda_{\bar{V}}(\xi) := \sum_{j=1}^{S} \alpha_j \bar{V}((p_j,\iota))$. Let $\bar{\Upsilon} := \bar{U} \times \mathbb{I}$ and $\bar{\mathscr{V}}_L$ be the set of L-Lipschitz real valued functions on $\bar{\Xi}$ and define $\bar{T} : \bar{\mathscr{V}}_L \to \bar{\mathscr{V}}_L$ as follows: $(\forall \bar{V} \in \bar{\mathscr{V}}_L)$ $(\forall \xi \in \bar{\Xi})$

$$\bar{T}[\bar{V}](\xi) := \min_{\upsilon \in \bar{\Upsilon}} \{\beta \Lambda_{\bar{V}}(\phi(\xi, \upsilon)) + \ell(\xi, \upsilon)\} \,. \qquad (5)$$

The following proposition, proved in the Appendix together with some other preliminary results, shows that the fixed point of $\bar{T}$ is a good approximation of the fixed point of T.

Proposition II.3. *Let $\bar{V}^*$ and V^* be the fixed points of operators $\bar{T}$ and T, respectively. Then, there exists a positive constant C such that*

$$(\forall \xi \in \bar{\Xi}) \ |\bar{V}^*(\xi) - V^*(\xi)| \leq C(\delta_u + \delta_x).$$

Since $|\bar{\Xi}| < \infty$, there exists a bijective function $h : \bar{\Xi} \to \{1, \dots, |\bar{\Xi}|\}$ that assigns to each element of $\bar{\Xi}$ a natural number between 1 and $|\bar{\Xi}|$. Define a linear function $\Pi : \bar{\mathscr{V}}_L \to \mathbb{R}^{|\bar{\Xi}|}$, that associates to a function $\bar{V} \in \bar{\mathscr{V}}_L$ a vector $\Pi(\bar{V}) \in \mathbb{R}^{|\bar{\Xi}|}$, such that, $(\forall \xi \in \bar{\Xi})$ $[\Pi(\bar{V})]_{h(\xi)} = \bar{V}(\xi)$. Note that Π is invertible.

Now, we will show that $\bar{V}^*$ can be computed as the solution of a problem with the following form:

$$\begin{aligned} &\max_{\pi} F(\pi) \\ \text{subject to} \quad & 0 \leq \pi \leq G(\pi), \ \pi \leq \mathscr{U}, \end{aligned} \qquad (6)$$

where, $\pi \in \mathbb{R}^{|\bar{\Xi}|}$, $F : \mathbb{R}^{|\bar{\Xi}|} \to \mathbb{R}$ is any strictly monotone increasing function with respect to each component, $\mathscr{U} \in \mathbb{R}^{|\bar{\Xi}|}$ is a real constant vector and $G = (G_1, \dots, G_{|\bar{\Xi}|}) : \mathbb{R}^{|\bar{\Xi}|} \to \mathbb{R}^{|\bar{\Xi}|}$, is a continuous function such that, for $i \in \{1, \dots, |\bar{\Xi}|\}$, G_i is monotone not decreasing with respect to all variables, constant with respect to $[\pi]_i$ and with the following form:

$$G(\pi) = \bigwedge_{\mathfrak{l} \in \mathscr{L}} \{A_{\mathfrak{l}} \pi + b_{\mathfrak{l}}\} \,, \qquad (7)$$

where $\mathscr{L}$ is a finite set and for each $\mathfrak{l} \in \mathscr{L}$, $A_{\mathfrak{l}}$ is a nonnegative and sparse matrix and $b_{\mathfrak{l}}$ is a nonnegative vector, and $\bigwedge$ represents the greatest lower bound operator. Note that $\mathscr{U}$ represents an upper bound for the solution of (6) that needs to be introduced since it will be used by the solution algorithm.

Proposition II.4. *Let $\bar{V}^*$ be the solution of* (5) *and set $\pi^* = \Pi(\bar{V}^*)$, then π^* solves a problem of form* (6)–(7).

Proof. Let $r \in \{1, \dots, |\bar{\Xi}|\}$, $\xi_r := h^{-1}(r)$. Since $\bar{V}^*(\xi_r) = \bar{T}[\bar{V}^*](\xi_r)$, $[\pi^*]_r = [\Pi(\bar{V}^*)]_r = \bar{V}^*(\xi_r) = \bar{T}[\bar{V}^*](\xi_r)$, that is, by (5), $[\pi^*]_r = \min_{\upsilon \in \bar{\Upsilon}} \left\{ \beta \Lambda_{\Pi^{-1}(\pi^*)}(\phi(\xi_r, \upsilon)) + \ell(\xi_r, \upsilon) \right\}$. Let $\upsilon = (u, \iota) \in \bar{\Upsilon}$ and $\{p_{r_1}, \dots, p_{r_S}\} \subset \bar{X}$, be the extremal points of the simplex of the triangulation containing ξ_r, then there exist coefficients $\alpha_{r_j} = \alpha_{r_j}(\upsilon) \in [0,1]$, with $j \in \{1, \dots, S\}$, such that $\sum_{j=1}^S \alpha_{r_j} = 1$ and $\phi(\xi_r, \upsilon) = \sum_{j=1}^S \alpha_{r_j} p_{r_j}$. Hence, $\Lambda_{\Pi^{-1}(\pi^*)}(\phi(\xi_r, \upsilon)) = \Lambda_{\bar{V}^*}(\phi(\xi_r, \upsilon)) = \sum_{j=1}^S \alpha_{r_j} \bar{V}^*(p_{r_j}, \iota) = \sum_{j=1}^S \alpha_{r_j} \bar{T}[\bar{V}^*](p_{r_j}, \iota)$. Then, recalling that $\beta \in (0,1)$, we have

$$[\pi^*]_r = \min_{\upsilon \in \bar{\Upsilon}} \left\{ \beta \sum_{j=1}^S \alpha_{r_j} [\pi^*]_{h((p_{r_j}, \iota))} + \ell(\xi_r, \upsilon) \right\} \Leftrightarrow$$

$$(1 - \beta \alpha_r)[\pi^*]_r = \min_{\upsilon \in \bar{\Upsilon}} \left\{ \beta \sum_{j=1, r_j \neq r}^S \alpha_{r_j} [\pi^*]_{h((p_{r_j}, \iota))} + \ell(\xi_r, \upsilon) \right\} \Leftrightarrow$$

$$[\pi^*]_r = \frac{1}{1 - \beta \alpha_r} \min_{\upsilon \in \bar{\Upsilon}} \left\{ \beta \sum_{j=1, r_j \neq r}^S \alpha_{r_j} [\pi^*]_{h((p_{r_j}, \iota))} + \ell(\xi_r, \upsilon) \right\}.$$

For any $j \in \{1, \dots, S\}$, $r, r_j \in \{1, \dots, |\bar{\Xi}|\}$ and $\upsilon \in \bar{\Upsilon}$, set

$$[A_\upsilon]_{r, r_j} := \begin{cases} \frac{\beta \alpha_{r_j}(\upsilon)}{1 - \beta \alpha_r(\upsilon)}, & r_j \neq r \\ 0, & r_j = r \end{cases} \quad [b_\upsilon]_r := \frac{\ell(\xi_r, \upsilon)}{1 - \beta \alpha_r(\upsilon)}. \quad (8)$$

Note that A_υ and b_υ, with $\upsilon \in \bar{\Upsilon}$, are nonnegative. Moreover, matrices A_υ are sparse since, for any $r \in \{1, \dots, |\bar{\Xi}|\}$, there are at most $S \ll |\bar{\Xi}|$ nonzero entries in the r-th row of A_υ. Then, we obtain that

$$\pi^* = \min_{\upsilon \in \bar{\Upsilon}} \{A_\upsilon \pi^* + b_\upsilon\}. \qquad (9)$$

By Proposition 3.4 of [35], the solution of (9) corresponds to the optimal solution of a problem of form (6), in which F is any strictly monotone increasing function (for instance, $F := \|\cdot\|_1$), $G(\pi) = \bigwedge_{\upsilon \in \bar{\Upsilon}} \{A_\upsilon \pi + b_\upsilon\}$ and $\mathscr{U} = \max_{r \in \{1 \dots, |\bar{\Xi}|\}, \upsilon \in \bar{\Upsilon}} \frac{\ell(\xi_r, \upsilon)}{1 - \beta \alpha_r(\upsilon)}$. $\square$

It is natural to associate to Problem (6) a directed graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where nodes correspond to the $|\bar{\Xi}|$ components of π and of constraint $G(\pi) = \bigwedge_{\mathfrak{l} \in \mathscr{L}} \{A_{\mathfrak{l}} \pi + b_{\mathfrak{l}}\}$, namely $\mathbb{V} = \bar{\Xi} \cup \mathscr{C}$, with $\bar{\Xi} = \{\xi_1, \dots, \xi_{|\bar{\Xi}|}\}$, $\mathscr{C} = \{c_1, \dots, c_{\bar{\Xi}}\}$, where v_i is the node associated to $[\pi]_i$ and c_i is the node associated to G_i, for $i \in \{1, \dots, |\bar{\Xi}|\}$. Whilst, the edge set $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$ is defined according to the following rules:

- for $i \in \{1, \dots, |\bar{\Xi}|\}$, there is an edge from c_i to ξ_i,
- for $i, j \in \{1, \dots, |\bar{\Xi}|\}$, there is an edge from ξ_i to c_j if G_j depends on $[\pi]_i$,
- no other edges are present in $\mathbb{E}$.

III. Solution method

We define an approximated solution of Problem (6) as follows: let ε be a positive real constant, we call π an ε-solution of (6) if $\pi \geq a$ and $\|\pi - G(\pi)\| < \varepsilon$.

Note that, since function G of (7) with matrices $A_{\mathfrak{l}}$ and vectors $b_{\mathfrak{l}}$ defined as in (8) is a contraction, it admits a unique fixed point. Thus, an ε-solution can be found with a standard fixed point iteration. In [35], we proposed an algorithm which exploits the special structure of Problem (6) and is more efficient than a simple fixed point iteration. Note that Problem (6) with G of (7)–(8) is a Linear Programming problem which could be solved by means of any linear solver. However, in [35] we already showed how a commercial solver like Gurobi is outperformed by the algorithm introduced in [35] (we refer the reader to [35] for a more in-depth discussion). Note that the same problem is related to the one tackled in [42], [43]. The algorithm proposed in these works has polynomial complexity but only works under an assumption (absence of cycles of decreasing weight) which is not fulfilled in our case. Also [44] addresses a similar problem, however the superiority condition and its generalizations introduced there do not hold in our case.

The order in which nodes are actually processed depends on the ordering of the priority queue. The choice of this ordering turns out to be critical in terms of computational cost for the algorithm, as can be seen in the numerical experiments in Section VI. The procedure stops once the priority queue becomes empty, that is, once none of the updated nodes undergoes a significant variation. The correctness of the algorithm is independent on the choice of the ordering of the priority queue.

IV. An application to path planning

We return to the car-like model presented in the Introduction. The relation between the angular velocity ω and the front wheel steering angle δ is given by $\omega = \frac{1}{l} v \tan\delta$, where l is the distance between the front and rear axles of the car. The input control variables are constrained as follows: $v_{\min} < v < v_{\max}$ and $\delta_{\min} < \delta < \delta_{\max}$, with $v_{\min} < 0$ and $v_{\max} > 0$. The state space is given by a bounded and connected domain $\Omega \subseteq \mathbb{R}^2 \times S^1$, partitioned into two spaces: Ω_{fr}, representing the free space, and Ω_{ob}, representing the space occupied by obstacles. These two spaces form a partition of Ω, that is, they are such that $\Omega_{\text{fr}} \cup \Omega_{\text{ob}} = \Omega$, and $\Omega_{\text{fr}} \cap \Omega_{\text{ob}} = \varnothing$.
In cost function (3), we set $(\forall x \in X)\ (\forall u \in U)$

$$\ell_1(x,u) = \begin{cases} 1, & \text{if } x \in \Omega_{\text{fr}} \\ \frac{1}{1-\beta}, & \text{if } x \in \Omega_{\text{ob}}\,. \end{cases}$$

In the following, we will present four finite state machines for addressing the path planning problem.

Model 1. The first finite state machine, depicted in Figure 1, is $\mathbb{A}_1 = (\mathbb{I}_1, \Sigma_1, \rho_1, \mathrm{F}, \mathbb{I}_1)$ in which $\mathbb{I}_1 = \{\mathrm{F},\mathrm{B}\}$, with F and B representing the control subsystem in which the car-like vehicle is moving in forward direction and, respectively, in backward direction; $\Sigma_1 = \{c\}$, with symbol c representing a direction switching, w.l.o.g., the initial state is F, and all states are accept ones. The accepted language is $L(\mathbb{A}_1) = \Sigma_1^*$ with $\rho_1(\mathrm{F},c) = \mathrm{B}$ and $\rho_1(\mathrm{B},c) = \mathrm{F}$. For any $\iota \in \mathbb{I}_1$, we define the cost function associated to discrete state transitions as $\ell_2(\iota,c) = \bar{\ell}$, where $\bar{\ell} > 0$ is a penalization term associated to each change between forward and backward motion, and $\ell_2(\iota,\varepsilon) = 0$. In this way, a longer path with a lower number of direction changes can have a lower cost than a shorter one with a larger number of direction changes. The system dynamic is defined as follows:

$$f(x,\iota,u) = \begin{cases} (\cos\theta, \sin\theta, \omega)^T, & \text{if } \iota = \mathrm{F} \\ (-\cos\theta, -\sin\theta, \omega)^T, & \text{if } \iota = \mathrm{B}. \end{cases} \tag{10}$$

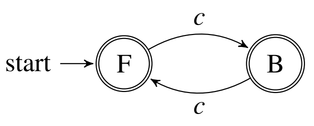


Fig. 1: Finite state machine $\mathbb{A}_1$.

Model 2. A second model considers finite state machine $\mathbb{A}_2 = (\mathbb{I}_2, \Sigma_2, \rho_2, K, \{0,\dots,K\})$, in which $\mathbb{I}_2 = \{0,\dots,K\} \cup \{\mathrm{I}\}$, where $K \in \mathbb{N}$ is the maximum number of allowed direction switchings, states $\{0,\dots,K\}$ represent the number of remaining allowed direction switchings and I is a reject state. The initial state is K, that is, the state in which the number of remaining allowed direction switchings is K, and the set of accept states is given by $\mathbb{I}_2 \setminus \{\mathrm{I}\}$ (that is, I is the the only reject state). The symbols set is $\Sigma_2 = \{c\}$, where the only symbol c is associated to a direction switching. The transition function is $\rho_2(i,c) = i-1$, for $i \in \{1,\dots,K\}$, and $\rho_2(i,c) = \mathrm{I}$, for $i \in \{0,\mathrm{I}\}$. In this way, the accepted language is $L(\mathbb{A}_2) = \{\varepsilon, c, c^2, \dots, c^K\}$. Figure 2 represents finite state machine $\mathbb{A}_2$. As in the previous model, here we have only two control subsystems representing the car-like vehicle moving in forward or backward direction. If $x = (w,y,\theta)$, the system dynamics (1) are defined as follows

$$f(x,\iota,u) = \begin{cases} (\cos\theta, \sin\theta, \omega)^T, & \text{if } \iota \neq \mathrm{I} \text{ and even} \\ (-\cos\theta, -\sin\theta, \omega)^T, & \text{if } \iota \neq \mathrm{I} \text{ and odd} \\ (0,0,0), & \text{if } \iota = \mathrm{I}. \end{cases} \tag{11}$$

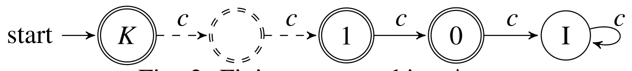


Fig. 2: Finite state machine $\mathbb{A}_2$.

Model 3. The third finite state machine is defined as $\mathbb{A}_3 = (\mathbb{I}_3, \Sigma_3, \rho_3, \mathrm{Bc}, \mathbb{I}_3)$, in which $\mathbb{I}_3 = \{\mathrm{Fl}, \mathrm{Fc}, \mathrm{Fr}, \mathrm{Bl}, \mathrm{Bc}, \mathrm{Br}\}$ and the system dynamics are

$$f(x,\iota,u) = \begin{cases} (\cos\theta, \sin\theta, \omega_{\min})^T, & \text{if } \iota = \mathrm{Fl} \\ (\cos\theta, \sin\theta, 0)^T, & \text{if } \iota = \mathrm{Fc} \\ (\cos\theta, \sin\theta, \omega_{\max})^T, & \text{if } \iota = \mathrm{Fr} \\ (-\cos\theta, -\sin\theta, \omega_{\min})^T, & \text{if } \iota = \mathrm{Bl} \\ (-\cos\theta, -\sin\theta, 0)^T, & \text{if } \iota = \mathrm{Bc} \\ (-\cos\theta, -\sin\theta, \omega_{\max})^T, & \text{if } \iota = \mathrm{Br}. \end{cases} \tag{12}$$

Namely, in state names, F and B denote states associated to forward and backward vehicle motion, respectively, whilst l, c and r denote left, straight and right steering, respectively. Moreover, $\omega_{\min} = -\frac{v}{l}\delta_{\max}$, $\omega_{\max} = \frac{v}{l}\delta_{\max}$, where $\delta_{\max}$ is the maximum allowed steering angle.

Any state of the machine is an accept one. The alphabet is given by $\Sigma_3 = \{\nwarrow, \uparrow, \nearrow, \swarrow, \downarrow, \searrow\}$. W.l.o.g., the initial state is Bc, that is, the state in which the vehicle is moving in backward direction and the steer is straight. Figure 3 represents the possible state transitions from state Bc; state transitions from other states are analogous to this one. The accepted language is $L(\mathbb{A}_3) = \Sigma_3^*$, and the transition function ρ_3 is defined as follows: $(\forall \iota \in \mathbb{I}_3)$ $\rho_3(\iota, \nwarrow) = \mathrm{Fl}$, $\rho_3(\iota, \uparrow) = \mathrm{Fc}$, $\rho_3(\iota, \nearrow) = \mathrm{Fr}$, $\rho_3(\iota, \swarrow) = \mathrm{Bl}$, $\rho_3(\iota, \downarrow) = \mathrm{Bc}$, $\rho_3(\iota, \searrow) = \mathrm{Br}$. For $\iota \in \mathbb{I}_3$, $\sigma \in \Sigma$, the cost function associated to the finite state machine transitions is $\ell_2(\iota, \sigma) = \bar{\ell}$, where $\bar{\ell} > 0$ is a penalization term associated to each motion change, and $\ell_2(\iota, \varepsilon) = 0$. Note that, as in the first model, even though there is no given limit over the number of direction changes, again, the transition cost function ℓ_2 provides an implicit bound over the number of state switchings.

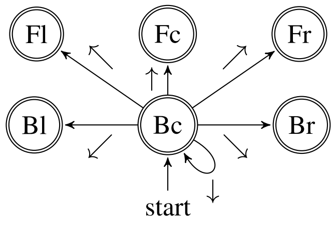


Fig. 3: State transitions from state Bc for $\mathbb{A}_3$.

Model 4. In the last model, we consider trajectories composed of a maximum of K segments. In each segment the input is constant, as in Model 3. Hence, in Model 4, the finite state machine is the parallel composition (see 2.3.2 of [45]) of $\mathbb{A}_2$ and $\mathbb{A}_3$. Namely, $\mathbb{A}_4 = (\mathbb{I}_2 \times \mathbb{I}_3, \Sigma_2 \cup \Sigma_3, \rho_4, (K, \mathrm{Bc}), \{0, \ldots, K\} \times \mathbb{I}_3)$. Transition function ρ_4 is such that $(\forall \iota \in \mathbb{I}_2)$ $(\forall \eta \in \mathbb{I}_3)$

$$\rho_4((\iota, \eta), \sigma) = \begin{cases} (\rho_2(\iota, \sigma), \eta), & \text{if } \sigma \in \Sigma_2 \\ (\iota, \rho_3(\eta, \sigma)), & \text{if } \sigma \in \Sigma_3 \,. \end{cases}$$

The system dynamics are defined as $f_4(x, (\iota, \eta), u) = f(x, \eta, u)$, with f defined as in (12).

V. Strongly connected components decomposition

In some cases, the queue policy can take into account a possible ordering of the states of the finite state machine. For instance, in the finite state machine associated to Model 2, the states are traversed in the order $K, K-1, \ldots, 0$. For this reason, the value of the variables associated to state K depend on $K-1$, but the converse is not true. Due to this structure, it makes sense to compute first the variables associated to state 0, then state 1, and proceed in this order up to state K (the same reasoning applies also to Model 4). In this section we present a general approach for developing a strategy for efficiently solving Problem (6), in case the discrete states exhibit an ordering. Here, we directly consider graph $\mathbb{G}$, associated to Problem (6) and introduced in Section II. Note that graph $\mathbb{G}$ already carries the information about finite state machine $\mathbb{A}$ allowed transitions. Our goal is to partition graph $\mathbb{G}$ in such a way that the obtained partition allows for a more efficient graph traversal when solving Problem (6). In order to do so, we introduce the following binary relation $\sim$ on nodes of $\mathbb{G}$. Given $v, w \in \mathbb{V}$ we say that $v \sim w$ if there is a directed path that connects v to w and viceversa. Relation $\sim$ is an equivalence one, hence, it induces a partition of graph $\mathbb{G}$ into equivalence classes which are subgraphs of $\mathbb{G}$ that are called strongly connected components (SCCs). Thus, we compute the SCCs of graph $\mathbb{G}$ and reduce each one of them into a single node, obtaining a new graph $\mathbb{C}$ which is called the condensation of $\mathbb{G}$. More formally, $\mathbb{C} = (\mathbb{W}, \mathbb{A})$ is a quotient graph in which $\mathbb{W} := \mathbb{V}/\sim$ and $\mathbb{A}$ is the edge set defined as follows $\mathbb{A} := \{a \in \mathbb{W} \times \mathbb{W} \mid a = ([w], [w']) \wedge (\exists v, v' \in \mathbb{V})\ v \not\sim v' \wedge v \in [w] \wedge v' \in [w'] \wedge (v, v') \in \mathbb{E}\}$, that is, there is an edge from a node $[w]$ to another one $[w']$ in condensation graph $\mathbb{C}$ only if a node of SCC $[w]$ in $\mathbb{G}$ has an edge in $\mathbb{E}$ that connects it to a node of SCC $[w']$ in $\mathbb{G}$. Graph $\mathbb{C}$ is known to be a directed acyclic graph (DAG). DAGs naturally carry a partial order on their nodes based on the concept of reachability. However, any DAG always admits, at least, one total order (see, for instance, [46]); in other words, there is at least one way for extending the partial order on $\mathbb{C}$ to a total order (see, for instance, [47]). This information on the total order on the nodes of $\mathbb{C}$ allows us to avoid many unnecessary computations when solving Problem (6). We solve the subproblem associated to the least (in the total order on $\mathbb{C}$) SCC $\mathbb{G}'$ of $\mathbb{G}$, using a priority queue policy. Note that SCC $\mathbb{G}'$ contains only those edges connecting nodes within it. This means that, when solving the subproblem associated to $\mathbb{G}'$, we are avoiding to perform all those updates due to edges going from a node of $\mathbb{G}'$ to a node outside of it. Once an ε-solution of the subproblem has been computed, we consider all those nodes in $\mathbb{G}$ and not in $\mathbb{G}'$ that are reachable from a node of $\mathbb{G}'$ through one edge and update their value. Then, we move to the next SCC given by the total order on $\mathbb{C}$ and proceed, as we just explained, until the greatest element of the total order is processed.

The advantages of this algorithm with respect to the state order corrected policy introduced in [38] are that the graph partitioning is automated and does not require any knowledge of the finite state machine $\mathbb{A}$ for performing the partitioning and it can produce a finer partitioning with respect to the introduced in [38], allowing for a further reduction of unnecessary node updates.

There are efficient algorithms for computing the SCCs of a graph, the Tarjan's algorithm (see [48]), among others, has a linear time complexity with respect to the number of edges and nodes of the graph.

Another way for exploiting the condensation of $\mathbb{G}$ is to consider the partial order $\leq$ on $\mathbb{C}$ and use it for parallelizing the computation of the solution of those SCCs that are not comparable to each other according to the partial order, that is, of those $[v], [w] \in \mathbb{C}, [v] \neq [w]$ such that $[v] \not\leq [w]$, and $[v] \not\leq [w]$, allowing for an overall decrease of the computational time for solving Problem (6).

VI. Numerical experiments

We consider three urban scenarios: a cross parking one (S1), in which a vehicle has to back into a perpendicular parking lot; a parallel parking one (S2), in which a vehicle has to park into a parallel parking lot; and a turn-around one (S3) in which a vehicle has to perform a 180° turn in a narrow space. The geometry of these scenarios are taken from real-life parking lots. For each scenario, we computed the optimal path with the 4 models presented in Section IV. For each model, we used the algorithm introduced in [35] for solving them using FIFO policy and largest node variation policy (i.e., first update nodes with largest error) for the priority queue. For Models 2 and 4, we also used the SCC decomposition coupled with FIFO and largest node variation policies. For these simulations, we considered a vehicle of the size of a typical B-segment passenger car with a length of 4.053 m and a width of 1.751 m. We discretized the state space using a grid with 31 334 vertices for S1, 57 844 vertices for S2, and 63 274 for S3. Moreover, the transition function of finite state machines associated to Models 2 and 4 are modeled with a limited number of direction changes fixed to $K=8$ for S1, to $K=10$ for S2 and to $K=12$ for S3 and with a switching penalty $\bar{\ell}=10$. Tests were run on a 2.7 GHz Intel Core i5 dual-core with 8 GB of RAM and the algorithm has been implemented in C++ We computed the numerical solution of the switching Bellman equation and the computational time results are reported in Table I. The table shows how the solution time varies according to which finite state machine is used for modeling the switchings and to which priority queue policy is adopted for solving the problem.

As we can see from the table, neither of the proposed policies prevails in terms of computational times for all the four introduced models. However, for Models 1 and 3 it seems that the largest node variation policy provides better computational times, whilst, on the contrary, for Models 2 and 4 it seems that the SCC decomposition coupled with the FIFO policy is faster. Note that, in this work we omitted to introduce other policies that we proved being outperformed by the FIFO and largest node variation policies in an earlier work (see [35]). Moreover, as we already pointed out in Section III, a commercial solver for linear problems like Gurobi is strongly outperformed by the algorithm used in this section.

Note that the computational times shown in Table I refer to solving the switching Bellman equation on the whole discretized state space whilst the time needed for computing a path from a given initial position is negligible as it is of the order of milliseconds. This is also why we do not present comparisons in terms of computational times with other methods. Since they all aim at computing one single path given a specific initial position, as opposed to our approach, which allows obtaining a path for any initial position of the scenario, the juxtaposition of such results would be difficult to interpret. Besides, note that the best computational times in Table I are compatible with the waiting time of a final user of an autonomous vehicle from the moment a parking space is detected to the moment the vehicle starts performing the parking maneuver. Moreover, parallelizing the algorithm from [35] exploiting equivalence relation $\sim$ introduced in Section V, together with further optimizations of the code, could lead to better computational times. We also generated several paths starting from different initial positions of the vehicle in the parking scenario. In Figure 4, we present some of them to give an idea of the generated paths from a qualitative point of view. In each subfigure, the green vehicle represents the final target position, the magenta vehicle represents the initial position, the black and red lines represent the generated path with black representing forward motion and red representing backward motion, the blue vehicle represents the attained final position, the red area represents the obstacles of the scenario, the small white box containing the green vehicle represents the parking lot, the one containing the magenta vehicle represents the roadway, whilst the wider red box represents the boundaries of the parking scenario. Comparisons of the obtained paths from the qualitative point of view can be done with respect to [11], which considers similar scenarios. In particular, the obtained paths are qualitatively similar to those presented in [11]. In fact, our method allows to obtain paths that have a simpler structure, that is, a smaller number of direction changes, with respect to RRT on all the three presented scenarios and also with respect to SEHS on the turn-around scenario of [11].

Model	Policy	S1	S2	S3
1	FIFO	4.28 s	5.33 s	6.92 s
2	SCC + FIFO	1.82 s	2.51 s	6.08 s
3	FIFO	4.22 s	7.23 s	8.80 s
4	SCC + FIFO	2.60 s	6.27 s	11.30 s
1	variation	1.14 s	2.78 s	3.51 s
2	SCC + var.	2.67 s	5.52 s	13.90 s
3	variation	1.95 s	4.37 s	7.20 s
4	SCC + var.	2.64 s	8.14 s	13.97 s

TABLE I: Computational times for solving the problem on the 3 scenarios using the 4 models and priority policies.

Appendix

Here we present the proof of Proposition II.3 starting from some instrumental results. Note that the material presented here is an adaptation to our setting of standard arguments on the solution of Bellman equation (see [41]) and its numerical solution (see [33]).

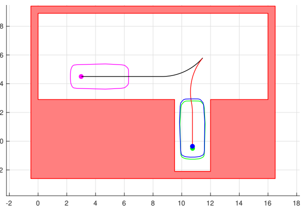
By Assumption II.2, note that $\exists L_{f_x}, L_{f_u}, L_{\ell_x}, L_{\ell_u} \geq 0$ such that $(\forall x_1, x_2 \in X)$ $(\forall \iota \in \mathbb{I})$ $(\forall u \in U)$ $(\forall \sigma \in \Sigma \cup \{\varepsilon\})$

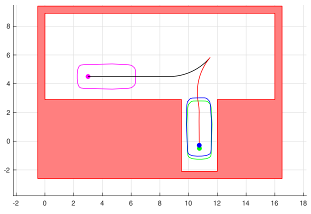
$$\|f(x_1,\iota,u)-f(x_2,\iota,u)\| \leq L_{f_x}\|x_1-x_2\|,$$
$$|\ell((x_1,\iota),(u,\sigma))-\ell((x_2,\iota),(u,\sigma))| \leq L_{\ell_x}\|x_1-x_2\|,$$

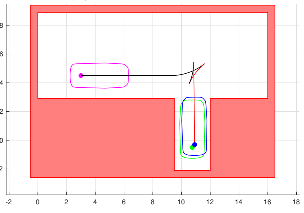
$(\forall x \in X)$ $(\forall \iota \in \mathbb{I})$ $(\forall u_1, u_2 \in U)$ $(\forall \sigma \in \Sigma \cup \{\varepsilon\})$

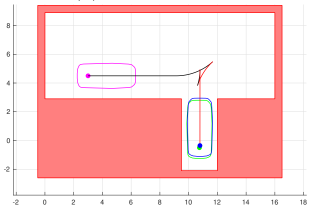
$$\|f(x,\iota,u_1)-f(x,\iota,u_2)\| \leq L_{f_u}\|u_1-u_2\|,$$
$$|\ell((x,\iota),(u_1,\sigma))-\ell((x,\iota),(u_2,\sigma))| \leq L_{\ell_u}\|u_1-u_2\|.$$

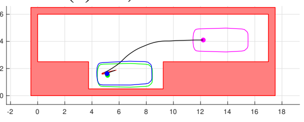
In the following proposition and corollary, we show that, under certain conditions, set $\mathscr{V}_L$ is invariant under T defined as in (4).

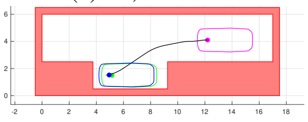

(a) S1, Model 1.

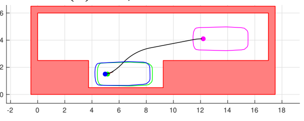

(b) S1, Model 2.

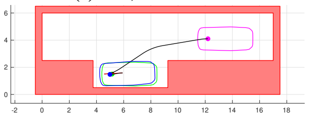

(c) S1, Model 3.

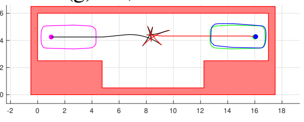

(d) S1, Model 4.

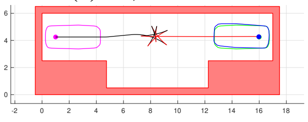

(e) S2, Model 1.

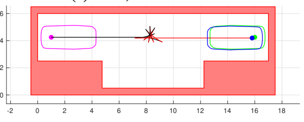

(f) S2, Model 2.


(g) S2, Model 3.


(h) S2, Model 4.


(i) S3, Model 1.


(j) S3, Model 2.


(k) S3, Model 3.

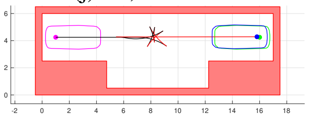

(l) S3, Model 4.

Fig. 4: Paths obtained from the 4 models over 3 scenarios.

Proposition A.1. *$\beta L_{f_x} < 1 \Rightarrow (\forall L > 0) T[\mathscr{V}_L] \subseteq \mathscr{V}_{\beta L L_{f_x} + L_{\ell_x}}$.*

Proof. Let $V \in \mathscr{V}_L$, $\iota \in \mathbb{I}$ and $\xi_1 = (x_1, \iota), \xi_2 = (x_2, \iota) \in \Xi$. Let assume wlog that $T[V](\xi_1) \geq T[V](\xi_2)$ and let $\upsilon^* = (u^*, \iota^*) := \text{argmin}_{\upsilon \in \Upsilon}\{\beta V(\phi(\xi_2, \upsilon)) + \ell(\xi_2, \upsilon)\}$, that is well defined since Υ is compact. Then $|T[V](\xi_1) - T[V](\xi_2)| =$

$$= |\min_{\upsilon\in\Upsilon}\{\beta V(\phi(\xi_1,\upsilon)) + \ell(\xi_1,\upsilon)\} - \beta V(\phi(\xi_2,\upsilon^*)) - \ell(\xi_2,\upsilon^*)|$$
$$\leq \beta|V(\phi(\xi_1,\upsilon^*)) - V(\phi(\xi_2,\upsilon^*))| + |\ell(\xi_1,\upsilon^*) - \ell(\xi_2,\upsilon^*)|$$
$$\leq \beta L\|\phi(\xi_1,\upsilon^*) - \phi(\xi_1,\upsilon^*)\| + |\ell(\xi_1,\upsilon^*) - \ell(\xi_2,\upsilon^*)|$$
$$= \beta L\|f((x_1,\iota),u^*) - f((x_2,\iota),u^*)\| +$$
$$+ |\ell((x_1,\iota),\upsilon^*) - \ell((x_2,\iota),\upsilon^*)| \leq (\beta L L_{f_x} + L_{\ell_x})\|x_1 - x_2\|. \quad \square$$

Corollary A.2. *If $L \geq \frac{L_{\ell_x}}{1-\beta L_{f_x}}$, then $T[\mathscr{V}_L] \subseteq \mathscr{V}_L$.*

Now, the Bellman equation associated to system (2) and cost function (3) is given by $V = T[V]$. The following proposition shows that T is a contractive mapping.

Proposition A.3. *If $V_1, V_2 \in \mathscr{V}_L$, then $\|T[V_1] - T[V_2]\| \leq \beta\|V_1 - V_2\|$.*

Proof. Let $\xi \in \Xi$, w.l.o.g., $T[V_1](\xi) \geq T[V_2](\xi)$ and let $\upsilon^* := \text{argmin}_{\upsilon\in\Upsilon}\{\beta V_2(\phi(\xi,\upsilon)) + \ell(\xi,\upsilon)\}$, then $(T[V_1] - T[V_2])(\xi) \leq \beta(V_1 - V_2)(\phi(\xi,\upsilon^*)) \leq \beta\|V_1 - V_2\|$. $\square$

Since $(\mathscr{V}_L, \|\cdot\|)$ is a complete normed space (see Lemma 8.1.4 in [49]), by the Banach-Caccioppoli fixed-point theorem, T has a unique fixed point, denoted by V^*, that corresponds to the solution of Bellman equation.

Similarly as before, one can prove that $\bar{T}$ defined as in (5) is a contraction on $\bar{\mathscr{V}}_L$ and denote by $\bar{V}^*$ its fixed point. The following proposition serves to show that $\bar{V}^*$ is a good approximation of V^*.

Proposition A.4. *There exists a positive constant C_1 such that $(\forall \xi \in \bar{\Xi})\ |\bar{T}[V^*|_{\bar{\Xi}}](\xi) - T[V^*](\xi)| \leq C_1(\delta_u + \delta_x)$, where $V^*|_{\bar{\Xi}}$ represents the restriction of V^* to $\bar{\Xi}$*

Proof. Let $\upsilon^* \in \Upsilon$ be the value corresponding to the minimum at $T[V^*](\xi)$, let $\bar{\upsilon}$ be the element in $\bar{\Upsilon}$ closest to υ^*, and let $\tilde{V}^* := V^*|_{\bar{\Xi}}$, then, if $\bar{T}[\tilde{V}^*](\xi) \geq T[V^*](\xi)$,

$$\bar{T}[\tilde{V}^*](\xi) - T[V^*](\xi) \leq$$
$$\leq \beta(\Lambda_{\tilde{V}^*}(\phi(\xi,\bar{\upsilon})) - V^*(\phi(\xi,\upsilon^*))) + \ell(\xi,\bar{\upsilon}) - \ell(\xi,\upsilon^*)$$
$$\leq \beta(\Lambda_{\tilde{V}^*}(\phi(\xi,\bar{\upsilon})) - \Lambda_{\tilde{V}^*}(\phi(\xi,\upsilon^*)) +$$
$$+ \Lambda_{\tilde{V}^*}(\phi(\xi,\upsilon^*)) - V^*(\phi(\xi,\upsilon^*))) + \ell(\xi,\bar{\upsilon}) - \ell(\xi,\upsilon^*).$$

Now, since f is Lipschitz and also $\Lambda_{\tilde{V}^*}$, since it is the convex combination of $\tilde{V}^* \in \bar{\mathscr{V}}_L$, we have that $|\Lambda_{\tilde{V}^*}(\phi(\xi,\bar{\upsilon})) - \Lambda_{\tilde{V}^*}(\phi(\xi,\upsilon^*))| \leq L L_{f_u}\delta_u$. By Theorem 4.1 (iii) of [50], we have that $|\Lambda_{\tilde{V}^*}(\phi(\xi,\upsilon^*)) - V^*(\phi(\xi,\upsilon^*))| \leq L L_{f_x}\delta_x$. Finally, by the lipschitzianity of ℓ, we have that $|\ell(\xi,\bar{\upsilon}) - \ell(\xi,\upsilon^*)| \leq L_{\ell_u}\delta_u$. Hence, by setting $C_1 := \max\{\beta L L_{f_u} + L_{\ell_u}, \beta L L_{f_x}\}$, we obtain $|\bar{T}[\tilde{V}^*](\xi) - T[V^*](\xi)| \leq C_1(\delta_u + \delta_x)$. Otherwise, if $\bar{T}[\tilde{V}^*](\xi) < T[V^*](\xi)$, let $\bar{\upsilon}^* \in \bar{\Upsilon}$ be the value that corresponds to the minimum at $\bar{T}[\tilde{V}^*](\xi)$, then $T[V^*](\xi) - \bar{T}[\tilde{V}^*](\xi) \leq \beta(V^*(\phi(\xi,\bar{\upsilon}^*)) - \Lambda_{\tilde{V}^*}(\phi(\xi,\bar{\upsilon}^*))) \leq \beta L L_{f_x}\delta_x \leq C_1(\delta_u + \delta_x)$. $\square$

Proof of Proposition II.3. Set $\tilde{V}^* := V^*|_{\bar{\Xi}}$, then $|\bar{V}^*(\xi) - V^*(\xi)| = |\bar{T}[\bar{V}^*](\xi) - T[V^*](\xi)| = |\bar{T}[\bar{V}^*](\xi) - \bar{T}[\tilde{V}^*](\xi) + \bar{T}[\tilde{V}^*](\xi) - T[V^*](\xi)|$. By Proposition A.4, it follows that $|\bar{V}^*(\xi) - V^*(\xi)| \leq |\bar{T}[\bar{V}^*](\xi) - \bar{T}[\tilde{V}^*](\xi)| + C_1(\delta_u + \delta_x)$. Now, by Proposition A.3 and by definition of $\tilde{V}^*$, we have that $|\bar{T}[\bar{V}^*](\xi) - \bar{T}[\tilde{V}^*](\xi)| \leq \beta|\bar{V}^*(\xi) - V^*(\xi)|$. Hence, $|\bar{V}^*(\xi) - V^*(\xi)| \leq C(\delta_u + \delta_x)$, with $C := \frac{C_1}{1-\beta}$. $\square$

References

[1] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 72–89, 1986.

[2] V. Muñoz, A. Ollero, M. Prado, and A. Simón, "Mobile robot trajectory planning with dynamic and kinematic constraints," in *Proc. of the 1994 IEEE Int. Conf. on Robotics and Automation*, vol. 4, San Diego, CA, May 1994, pp. 2802–2807.

[3] R. Solea and U. Nunes, "Trajectory planning with velocity planner for fully-automated passenger vehicles," in *IEEE Intelligent Transportation Systems Conference*, September 2006, pp. 474–480.

[4] J. Villagra, V. Milanés, J. Pérez, and J. Godoy, "Smooth path and speed planning for an automated public transport vehicle," *Robotics and Autonomous Systems*, vol. 60, pp. 252–265, 2012.

[5] A. Gupta and R. Divekar, "Autonomous parallel parking methodology for ackerman configured vehicles," *ACEEE International Journal on Communication*, vol. 1, no. 2, pp. 22–27, July 2010.

[6] R. Zou, S. Wang, Z. Wang, P. Zhao, and P. Zhou, "A reverse planning method of autonomous parking path," in *2020 5th Asia-Pacific Conference on Intelligent Robot Systems*, 2020, pp. 92–98.

[7] J. Kim, "Perpendicular parking of car-like robots allowing a cusp on the path," *IEEE Access*, 2020.

[8] M. F. Hsieh and Ü. Özguner, "A parking algorithm for an autonomous vehicle," in *2008 IEEE Intelligent Vehicles Symposium*, 2008, pp. 1155–1160.

[9] M. Brezak and I. Petrović, "Real-time approximation of clothoids with bounded error for path planning applications," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 507–515, April 2014.

[10] P. Petrov and F. Nashashibi, "Modeling and nonlinear adaptive control for autonomous vehicle overtaking," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1643–1656, August 2014.

[11] C. Chen, M. Rickert, and A. Knoll, "Path planning with orientation-aware space exploration guided heuristic search for autonomous parking and maneuvering," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015, pp. 1148–1153.

[12] X. Zhang, A. Liniger, A. Sakai, and F. Borrelli, "Autonomous parking using optimization-based collision avoidance," in *2018 IEEE Conference on Decision and Control*, vol. 4327–4332, 2018.

[13] G. Song and N. M. Amato, "Randomized motion planning for car-like robots with c-prm," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 37–42.

[14] Y. Wang, D. K. Jha, and Y. Akemi, "A two-stage rrt path planner for automated parking," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, August 2017, pp. 496–502.

[15] B. Lee, Y. Wei, and I. Y. Guo, "Automatic parking of self-driving car based on lidar," in *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, September 2017, pp. 241–246.

[16] K. Zheng and S. Liu, "Rrt based path planning for autonomous parking of vehicle," in *2018 IEEE 7th Data Driven Control and Learning Systems Conference*, May 2018, pp. 627–632.

[17] E. Hansen and Y. Wang, "Improving path accuracy for autonomous parking systems: An optimal control approach," in *2020 American Control Conference*, 2020, pp. 5243–5249.

[18] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, April 2016.

[19] G. Carbone and F. Gomez-Bravo, *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, ser. Mechanisms and Machine Science. Springer International Publishing, 2015.

[20] R. Takei and R. Tsai, "Optimal trajectories of curvature constrained motion in the Hamilton-Jacobi formulation," *Journal of Scientific Computing*, vol. 54, no. 2-3, pp. 622–644, 2013.

[21] T. C. Lee and Z. P. Jiang, "Uniform asymptotic stability of nonlinear switched systems with an application to mobile robots," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1235–1252, 2008.

[22] E. Hernandez-Vargas, P. Colaneri, R. Middleton, and F. Blanchini, "Discrete-time control for switched positive systems with application to mitigating viral escape," *International Journal of Robust and Nonlinear Control*, vol. 21, pp. 1093–1111, 2011.

[23] P. Colaneri, R. H. Middleton, Z. Chen, D. Caporale, and F. Blanchini, "Convexity of the cost functional in an optimal control problem for a class of positive switched systems," *Automatica*, vol. 50, pp. 1227–1234, 2014.

[24] C. Albea, G. Garcia, and L. Zaccarian, "Hybrid dynamic modeling and control of switched affine systems: application to DC-DC converters," in *2015 IEEE 54th Conference on Decision and Control*, Dec 2015, pp. 2264–2269.

[25] B. Niu, C. K. Ahn, H. Li, and M. Liu, "Adaptive control for stochastic switched nonlower triangular nonlinear systems and its application to a one-link manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 10, pp. 1701–1714, 2018.

[26] X. Xu and P. J. Antsaklis, "Optimal control of switched systems: New results and open problems," in *Proceedings of the American Control Conference*, June 2000, pp. 2683–2687.

[27] ——, "A dynamic programming approach for optimal control of switched systems," in *Proceedings of the 39th IEEE Conference on Decision and Control*, 2000, pp. 1822–1827.

[28] S. C. Bengea and R. A. DeCarlo, "Optimal control of switching systems," *Automatica*, vol. 41, pp. 11–27, 2004.

[29] C. Seatzu, D. Corona, A. Giua, and A. Bemporad, "Optimal control of continuous-time switched affine systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 5, pp. 726–741, May 2006.

[30] F. Zhu and P. J. Antsaklis, "Optimal control of hybrid switched systems: A brief survey," *Discrete Event Dynamic Systems*, vol. 25, pp. 345–364, 2015.

[31] I. Capuzzo Dolcetta and L. C. Evans, "Optimal switching for ordinary differential equations," *SIAM Journal on Control and Optimization*, vol. 22, no. 1, pp. 143–161, Jan 1984.

[32] H. Zhang and M. R. James, "Optimal control of hybrid systems and a system of quasi-variational inequalities," *SIAM journal on control and optimization*, vol. 45, no. 2, pp. 722–761, 2006.

[33] M. Falcone and R. Ferretti, *Semi-Lagrangian Approximation Schemes for Linear and Hamilton–Jacobi Equations*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2013.

[34] H. Zhang and M. R. James, "On computation of optimal switching HJB equation," in *Proceedings of the 45th IEEE Conference on Decision & Control*, 2006, pp. 2704–2709.

[35] L. Consolini, M. Laurini, and M. Locatelli, "Graph-based algorithms for the efficient solution of optimization problems involving monotone functions," *Computational Optimization and Applications*, vol. 73, pp. 101–128, 2019.

[36] M. Laurini, L. Consolini, and M. Locatelli, "A multigraph-based selective update method for the efficient solution of dynamic programming," in *2018 IEEE 57th Conference on Decision and Control*, Dec 2018, pp. 5916–5921.

[37] P. Micelli, L. Consolini, and M. Locatelli, "Path planning with limited numbers of maneuvers for automatic guided vehicles: an optimization-based approach," in *2017 25th Mediterranean Conference on Control and Automation*. IEEE, July 2017, pp. 204–209.

[38] M. Laurini, L. Consolini, and M. Locatelli, "Fast numerical solution of optimal control problems for switched systems: An application to path planning," in *2020 IEEE 59th Conference on Decision and Control (to appear)*, 2020.

[39] M. Sipser, *Introduction to the Theory of Computation*, 3rd ed. CENGAGE Learning, 2012.

[40] J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2nd ed. Pearson Education, 2001.

[41] T. Kamihigashi, "Elementary results on solutions to the bellman equation of dynamic programming: existence, uniqueness, and convergence," *Economic Theory*, vol. 56, pp. 251–273, 2014.

[42] G. Gallo, G. Longo, and S. Pallottino, "Directed hypergraphs and applications," *Discrete Applied Mathematics*, vol. 42, pp. 177–201, 1993.

[43] G. Gallo, C. Gentile, D. Pretolani, and G. Rago, "Max horn sat and the minimum cut problem in directed hypergraphs," *Mathematical Programming*, vol. 80, pp. 213–237, 1998.

[44] G. Ramalingam and T. Reps, "An incremental algorithm for a generalization of the shortest-path problem," *Journal of Algorithms*, vol. 21, no. 46, pp. 267–305, 1996.

[45] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, 2nd ed. Springer, 2008.

[46] R. Sedgewick and K. Wayne, *Algorithms*. Addison-Wesley, 2011.

[47] D. I. Spivak, *Category Theory for the Sciences*. MIT Press, 2014.

[48] R. E. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[49] Ştefan Cobzaş, R. Miculescu, and A. Nicolae, *Lipschitz Functions*, ser. Lecture Notes in Mathematics 2241. Springer, 2019.

[50] M. Stämpfe, "Optimal estimates for the linear interpolation error on simplices," *Journal of Approximation Theory*, vol. 103, pp. 78–90, 2000.