Systems & Control Letters 103 (2017) 50–57

Contents lists available at ScienceDirect

Systems & Control Letters

journal homepage: www.elsevier.com/locate/sysconle

An optimal complexity algorithm for minimum-time velocity planning

Luca Consolini *, Marco Locatelli, Andrea Minari, Aurelio Piazzi

Dipartimento di Ingegneria dell'Informazione, Università di Parma, Parco Area delle Scienze 181/a, 43124 Parma, Italy

ARTICLE INFO

Article history:
Received 13 April 2016
Received in revised form 3 February 2017
Accepted 5 February 2017

Keywords:
Optimization
Motion planning
Minimum-time problems
Hidden convexity

ABSTRACT

Velocity planning on a path to be followed by a wheeled autonomous vehicle may be difficult when high curvatures and velocities are allowed. A fast, straightforward algorithm to address this problem is presented. It has linear-time computational complexity and provides an optimal minimum-time velocity profile. The algorithm is based on a *curvilinear discretization* that makes easy to take into account the constraint on the vehicle's maximal normal acceleration. A generalized problem is also addressed with formal results on feasibility, complexity, and solution characterization. Three examples illustrate the proposed approach.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

The design and development of autonomous vehicles is a major technological advancement influencing and shaping our future mobility and quality of life [1]. Intense research is internationally carried out on the various technologies enabling the automated driving of wheeled autonomous vehicles [2]. In this context, focusing on motion planning, a significant problem is the *minimum-time* motion of a car-like vehicle from a start configuration to a target one while avoiding collisions (obstacle avoidance) and satisfying kinematic, dynamic and mechanical constraints (on velocities, accelerations, maximal steering angle, etc.). This problem can be then approached as: (1) a *minimum-time trajectory planning* where both the path to be followed by the vehicle and the timing law on this path (i.e., the vehicle's velocity) are simultaneously designed; or (2) a (geometric) *path planning* followed by a *minimum-time velocity planning* on the planned path.

In general, the latter is a sub-optimal approach with respect to the former. On the other hand, the latter which is a *path–velocity decomposition* [3] is easier to implement in real-time when the vehicle's motion supervisor uses a sensor-based iterative motion planning [4,5]. To comply with the vehicle's nonholonomic constraints, the planned path must have a certain geometric continuity and satisfy geometric interpolation conditions at its endpoints [5–7]. In turn, the velocity to be planned must comply with certain kinematic constraints (typically on velocities and accelerations) and interpolating boundary conditions.

An almost time-optimal velocity planning on a given path based on *ternary* polynomials is presented in [8]. A simple solution based on root extraction of a quartic equation is reported in [9] emphasizing arbitrary velocity/acceleration boundary conditions. A more elaborate solution based on *time discretization* is presented in [10]. It uses a sequence of linear programming feasibility tests to obtain a minimum-time velocity profile with constraints on velocities, accelerations and jerks. However, these works do not explicitly consider the constraint on the maximum normal (centripetal) acceleration of the vehicle.

This constraint is central to avoid the vehicle's sideslip especially when high curvatures and velocities may be present. An early work considering this constraint is [11]. An algorithm is then presented using linear profiles for the longitudinal accelerations. Other works addressing this issue are [12], [13], [14,15]. Solea and Nunes [12] achieve a solution with an algorithm based on the five-splines scheme of [16]. Villagra et al. [13] propose a closed-form speed profiler by using the elementary velocity profiles provided in [17]. The velocity planning of Chen et al. [14] is composed of simple linear or quadratic speed profiles. Li et al. [15] propose the generation of a velocity profile on the curvilinear frame (i.e., a velocity as a function of the path's arc-length) taking into account maximal normal and longitudinal accelerations and maximal brake decelerations. However, they do not formalize a minimum-time velocity planning problem.

In this paper a minimum-time velocity planning problem on a given path is proposed and solved. All the most relevant constraints are considered, namely: given bounds on maximum velocity, minimum and maximum longitudinal accelerations, and maximum

* Corresponding author.
E-mail addresses: luca.consolini@unipr.it (L. Consolini), marco.locatelli@unipr.it (M. Locatelli), andrea.minari2@studenti.unipr.it (A. Minari), aurelio.piazzi@unipr.it (A. Piazzi).

http://dx.doi.org/10.1016/j.sysconle.2017.02.001
0167-6911/© 2017 Elsevier B.V. All rights reserved.

normal acceleration. The interpolating boundary conditions are the vehicle's current velocity at the start of the path and the final velocity to be reached at the path's end.

A key point of the proposed solution is the *curvilinear discretization* on the path for which the sought velocity profile is a function of the arc-length (as in [15]). This discretization leads to a simpler nonlinear optimization problem with respect to the problem that would be obtained with the standard time discretization (see Section 2). An algorithm that provides a global solution to the problem is then presented (see Section 4). Remarkably, this algorithm has a computational time that increases linearly with the dimension n of the problem (being n the number of velocity samples on the path). Moreover, the algorithm during its three iterations (see (12)–(14)) requires very few computations and logical operations so that its implementation in real-time should be greatly facilitated. A theoretically interesting feature of the considered optimization problem is that the optimal solution corresponds to the component-wise maximum of the feasible set. This paper is an extended and revised version of [18].

Paper organization: Section 2 reports the motivation of the addressed velocity planning and the formal definition of the posed optimization problem (see Problem 2). This problem is nonlinear but has a hidden convexity highlighted in Section 3. A solution to Problem 2 provided by Algorithm 1 is reported in Section 4. It is a linear-time algorithm composed of three phases: a *forward iteration* (12), a *backward iteration* (13), and a final *join iteration* (14). Section 4 also presents a generalized problem (Problem 5) whose solution is provided by Algorithm 2. Related formal results on feasibility, complexity, and solution characterization are given by Theorems 1–3. Three examples are illustrated and discussed in Section 5. Concluding remarks are made in Section 6. Relevant propositions with proofs are reported in the final Section 7.

Notation: The set of real continuous functions with piecewise-continuous first-order derivative is denoted by C^1_{p}. A Cartesian path has first-order geometric continuity or is a G^1-path if it is continuous along the curvilinear abscissa with continuous unit tangent vector. A G^2-path, or a path with second-order geometric continuity, is a G^1-path with continuous curvature along the curvilinear abscissa. For more information on geometric continuity, see [19], [20]. For $a, b \in \mathbb{R}$, define $a \wedge b = \min\{a, b\}$, $a \vee b = \max\{a, b\}$, for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, define $\mathbf{u} \wedge \mathbf{v}$, $\mathbf{u} \vee \mathbf{v}$ as the analogous elementwise maximum and minimum operations. Define the partial order $\leq$ on $\mathbb{R}^n$ as

$$\mathbf{u} \leq \mathbf{w} \text{ if } u_i \leq w_i,\ i = 1, \ldots, n.$$

Given $U \subset \mathbb{R}^n$, we say that $\mathbf{u} \in U$ is the component-wise maximum of U if $\mathbf{u} \geq \mathbf{w}$, $\forall \mathbf{w} \in U$. Finally, define the extended real line $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\}$.

2. Motivation and problem formulation

In an automated driving scenario, a wheeled autonomous vehicle must follow a given Cartesian path starting from its current configuration to a future desired one (see Fig. 1). As known, the path must have continuous curvature (i.e., be a G^2-path) to admit a continuous steering function [5,6]. The arc length measured along the path is s and $k(s)$ is the path curvature as a function of s. Let s_{f} denote the total length of the path: this distance is requested to be traveled in minimum-time while satisfying constraints on the velocity and on the longitudinal and normal acceleration of the vehicle.

By denoting t_{f} as the time to travel the path, the problem can be approached by searching a velocity profile $v(t) \in C^1_{\mathrm{p}}[0, t_{\mathrm{f}}]$ as the solution of the following problem

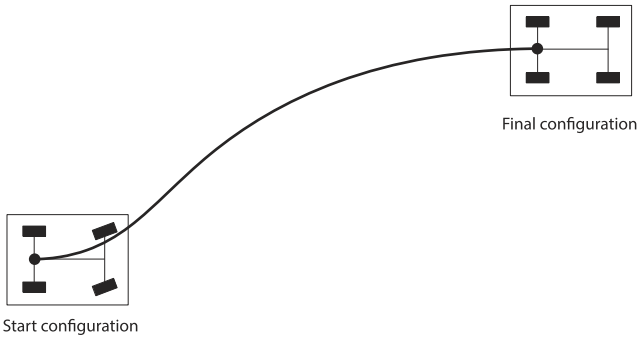

Fig. 1. A path to follow for an autonomous car-like vehicle.

Problem 1 (*Minimum-Time Problem in Continuous Time*)**.**

$$\min_{v \in C^1_{\mathrm{p}}[0, t_{\mathrm{f}}]} \quad t_{\mathrm{f}}$$

such that

$$s(t_{\mathrm{f}}) = s_{\mathrm{f}} \quad \text{with} \quad s(t) := \int_0^t v(\xi) d\xi,$$

$$v(0) = v_{\mathrm{s}}, \quad v(t_{\mathrm{f}}) = v_{\mathrm{f}}, \tag{1}$$

$$0 \leq v(t) \leq v_{\max},\ t \in [0, t_{\mathrm{f}}], \tag{2}$$

$$a^L_{\min} \leq \dot{v}(t) \leq a^L_{\max},\ t \in [0, t_{\mathrm{f}}], \tag{3}$$

$$v^2(t)|k(s(t))| \leq a^N_{\max},\ t \in [0, t_{\mathrm{f}}]. \tag{4}$$

Equalities (1) are the interpolation conditions where v_{s} is the vehicle's start velocity and v_{f} is the assigned final velocity at the path end. Inequalities (2) impose that the velocity cannot be negative (i.e., the vehicle cannot go backward along the planned path) and does not exceed the maximum value $v_{\max}$. Constraints (3) limit the maximum and minimum longitudinal accelerations by $a^L_{\min} < 0$ and $a^L_{\max} > 0$ whereas inequality (4) imposes the upper bound $a^N_{\max} > 0$ on the absolute value of the normal acceleration.

A difficulty in addressing Problem 1 lies in the normal acceleration constraint (4). In this form, it is a nonlinear constraint that makes difficult to find a global solution. With the aim to overcome this difficulty we propose a problem reformulation based on curvilinear discretization.

The path is uniformly divided into $n-1$ elementary path parts, each one of length $h := s_{\mathrm{f}}/(n-1)$. The endpoints of these elementary parts are denoted by $\mathbf{p}_i$, $i = 1, 2, \ldots, n$ ($\mathbf{p}_1$ and $\mathbf{p}_n$ are the starting and final points respectively). Let v_i be the vehicle's velocity at point $\mathbf{p}_i$. Then, the average speed on the ith path part (between $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$) is $(v_i + v_{i+1})/2$ and the corresponding time to travel this part is

$$t_i := \frac{2h}{v_i + v_{i+1}}.$$

The total time to travel the complete path is then

$$t_{\mathrm{f}} = \sum_{i=1}^{n-1} t_i = 2h \sum_{i=1}^{n-1} \frac{1}{v_i + v_{i+1}}.$$

The average longitudinal acceleration on the ith part of the path is

$$a^L_i := \frac{v_{i+1} - v_i}{t_i} = \frac{v^2_{i+1} - v^2_i}{2h}.$$

At point $\mathbf{p}_i$, let s_i be the arc length and the normal acceleration is then $v^2_i k(s_i)$. With this reformulation, the addressed planning is to find a velocity sequence $\mathbf{v} := (v_1, \ldots, v_n) \in \mathbb{R}^n$ to solve the

following optimization problem:

$$\min_{\mathbf{v}\in\mathbb{R}^n} 2h\sum_{i=1}^{n-1}\frac{1}{v_i+v_{i+1}}$$

such that

$$\begin{aligned}
&v_1 = v_s,\quad v_n = v_f\\
&0 \le v_i \le v_{\max},\ i=1,\dots,n\\
&a^L_{\min} \le a^L_i \le a^L_{\max},\ i=1,\dots,n-1,\\
&v_i^2|k(s_i)| \le a^N_{\max},\ i=1,\dots,n.
\end{aligned}$$

Define $k_i := k(s_i)$, $l_B := 2ha^L_{\min}$, $u_B := 2ha^L_{\max}$ (from the assumptions, note that $l_B < 0$ and $u_B > 0$ necessarily), and

$$\bar v_i := \min\left\{v_{\max}, \sqrt{a^N_{\max}/|k_i|}\right\},\ i=1,\dots,n.$$

This problem can be reformulated as follows.

Problem 2 (*Minimum-Time Velocity Planning Problem*)**.**

$$\min_{\mathbf{v}\in\mathbb{R}^n}\sum_{i=1}^{n-1}\frac{1}{v_i+v_{i+1}}$$

such that

$$v_1 = v_s,\quad v_n = v_f \tag{5}$$
$$0 \le v_i \le \bar v_i,\ i=1,\dots,n \tag{6}$$
$$l_B \le v_{i+1}^2 - v_i^2 \le u_B,\ i=1,\dots,n-1. \tag{7}$$

3. Hidden convexity

Problem 2 is a non-convex one. Indeed, constraints (7) are defined by indefinite quadratic functions. Non-convex problems are usually difficult to solve. However, there are classes of problems (the most famous being probably the class of *trust-region* problems where an indefinite quadratic function is minimized over an ellipsoid) which have a *hidden convexity* property. This means that the problems can be reformulated into equivalent convex problems. Problem 2 can be rewritten as:

$$\min_{\mathbf{v}\in\mathbb{R}^n}\sum_{i=1}^{n-1}\frac{1}{v_i+v_{i+1}}$$

such that

$$\begin{aligned}
&v_1 = v_s,\quad v_n = v_f,\\
&\frac{u_B}{v_i+v_{i+1}} \ge v_{i+1}-v_i,\ i=1,\dots,n-1\\
&\frac{-l_B}{v_i+v_{i+1}} \ge v_i - v_{i+1},\ i=1,\dots,n-1\\
&0 \le v_i \le \bar v_i,\ i=1,\dots,n.
\end{aligned}$$

Let $\mathbf{y}\in\mathbb{R}^{n-1}$ be a vector of auxiliary variables defined as

$$y_i = \frac{1}{v_i+v_{i+1}}\quad i=1,\dots,n-1.$$

Hence, Problem 2 can be reformulated as follows

Problem 3 (*Reformulation*)**.**

$$\min_{\mathbf{y}\in\mathbb{R}^{n-1},\mathbf{v}\in\mathbb{R}^n}\sum_{i=1}^{n-1} y_i$$

such that

$$\begin{aligned}
&v_1 = v_s,\quad v_n = v_f,\\
&u_B y_i \ge v_{i+1}-v_i,\ i=1,\dots,n-1\\
&-l_B y_i \ge v_i - v_{i+1},\ i=1,\dots,n-1\\
&y_i = \frac{1}{v_i+v_{i+1}},\ i=1,\dots,n-1\\
&0 \le v_i \le \bar v_i,\ i=1,\dots,n.
\end{aligned} \tag{8}$$

Equalities (8) can be relaxed into inequalities, so that we end up with the following convex relaxation of Problem 2:

Problem 4 (*Convex Relaxation of Problem 2*)**.**

$$\min_{\mathbf{y}\in\mathbb{R}^{n-1},\mathbf{v}\in\mathbb{R}^n}\sum_{i=1}^{n-1} y_i$$

such that

$$\begin{aligned}
&v_1 = v_s,\quad v_n = v_f,\\
&u_B y_i \ge v_{i+1}-v_i,\ i=1,\dots,n-1\\
&-l_B y_i \ge v_i - v_{i+1},\ i=1,\dots,n-1\\
&y_i \ge \frac{1}{v_i+v_{i+1}},\ i=1,\dots,n-1\\
&0 \le v_i \le \bar v_i,\ i=1,\dots,n.
\end{aligned}$$

In what follows we prove that the convex relaxation is exact, i.e., the optimal value of Problem 4 is equal to the optimal value of Problem 2, and, moreover, we can derive an optimal solution of Problem 2 from an optimal solution of Problem 4.

Theorem 1. *Problem 4 is an exact convex relaxation of Problem 2. In particular, there exists an optimal solution* $\mathbf{v}^*$, $\mathbf{y}^*$ *of Problem 4, such that* $\mathbf{v}^*$ *is an optimal solution of Problem 2.*

Proof. See Section 7.1.

In view of Theorem 1 we could solve Problem 2 through the convex Problem 4, which is polynomially solvable. However, in the next section we will propose an iterative linear-time procedure to solve Problem 2 which is more efficient than convex programming solvers.

4. The linear-time algorithm

In this section, we present an algorithm that solves Problem 2 in linear time with respect to the number of variables n. First, note that constraints (6), (7) can be rewritten as

$$v_i^2 - v_{i-1}^2 \le u_B,\ i=2,\dots,n \tag{9}$$
$$v_i^2 - v_{i+1}^2 \le -l_B,\ i=1,\dots,n-1 \tag{10}$$
$$\begin{aligned}
&v_i \le \bar v_i,\ i=1,\dots,n\\
&0 \le v_i,\ i=1,\dots,n.
\end{aligned} \tag{11}$$

The algorithm consists of a sequence of three iterations on variables v_i, $i=1,\dots,n$: the *forward iteration*, the *backward iteration* and the *join iteration*. The *forward iteration* computes a *forward solution* vector $\mathbf{f} := (f_1,\dots,f_n)\in\mathbb{R}^n$ according to

$$\begin{aligned}
&f_1 = v_s\\
&f_i = \min\left\{\sqrt{f_{i-1}^2+u_B}, \bar v_i\right\},\ i=2,\dots,n.
\end{aligned} \tag{12}$$

The initial condition f_1 is the velocity initial value v_s and f_i is set equal to the maximum speed such that the acceleration constraint (9) and the maximum velocity constraint (11) are satisfied

with $v_i = f_i$, $v_{i-1} = f_{i-1}$. In fact, the maximum velocity that satisfies (9) is given by $f_i = \sqrt{f_{i-1}^2 + u_B}$ and iteration (12) defines f_i as the minimum between this value and maximum velocity $\bar{v}_i$. The *backward iteration* is analogous but in reverse direction. It computes a velocity vector $\mathbf{b} := (b_1, \ldots, b_n) \in \mathbb{R}^n$ as the solution of

$$\begin{aligned} b_n &= v_f \\ b_i &= \min\left\{\sqrt{b_{i+1}^2 - l_B}, \bar{v}_i\right\}, \ i = n-1, \ldots, 1. \end{aligned} \tag{13}$$

The final condition b_n is set as the velocity final value v_f and the backward iteration (13) sets the value of b_i equal to the maximum speed such that the deceleration constraint (10) and the maximum velocity constraint (11) are satisfied with $v_i = b_i$, $v_{i+1} = b_{i+1}$. Namely, the maximum velocity that satisfies (10) is given by $\sqrt{b_{i+1}^2 - l_B}$ and iteration (13) sets b_i equal to the minimum between this value and maximum velocity $\bar{v}_i$. In the final step (*join iteration*), the optimal velocity profile is simply computed by

$$v_i^* = \min\{f_i, b_i\}, \ i = 1, \ldots, n. \tag{14}$$

Algorithm 1: Minimum-time velocity planning

input : v_s, v_f, u_B, l_B, $\bar{v}_i$, $i = 1, \ldots, n$
output: $\mathbf{v}^*$, *Feasible*

1 $f_1 \leftarrow v_s$
2 **for** $i \leftarrow 2$ **to** n **do**
3 $\quad f_i = \min\left\{\sqrt{f_{i-1}^2 + u_B}, \bar{v}_i\right\}$
4 $b_n \leftarrow v_f$
5 **for** $i \leftarrow n-1$ **to** 1 **do**
6 $\quad b_i = \min\left\{\sqrt{b_{i+1}^2 - l_B}, \bar{v}_i\right\}$
7 **for** $i \leftarrow 1$ **to** n **do**
8 $\quad v_i^* = \min\{f_i, b_i\}$
9 **if** $v_1^* = v_s$ *and* $v_n^* = v_f$ **then**
10 $\quad$ *Feasible* $\leftarrow$ *True*
11 **else**
12 $\quad$ *Feasible* $\leftarrow$ *False*

The overall procedure is reported in Algorithm 1. In particular, lines 1–3 implement the forward iteration, lines 4–6 the backward iteration and lines 7–8 the join iteration.

We claim that Algorithm 1 is able to check the feasibility of Problem 2 and that, in case the problem is feasible, vector $\mathbf{v}^*$ represents its optimal solution. Moreover, Algorithm 1 is an algorithm of optimal time-complexity.

Problem 2 belongs to the following class of problems.

Problem 5 (*Generalized Problem*)**.**

$$\min_{\mathbf{v}\in\mathbb{R}^n} f(\mathbf{v}) \tag{15}$$

such that

$$\mathbf{v}^- \le \mathbf{v} < \mathbf{v}^+ \tag{16}$$

$$g_i(v_{i-1}, v_i) \le 0, \ i = 2, \ldots, n \tag{17}$$

$$r_i(v_i, v_{i+1}) \le 0, \ i = 1, \ldots, n-1 \tag{18}$$

with the following assumptions:

- $f : \mathbb{R}^n \to \bar{\mathbb{R}}$ is a monotonic decreasing function (i.e., $\mathbf{v} \ge \mathbf{w}$ implies $f(\mathbf{v}) \le f(\mathbf{w})$).
- For $i = 2, \ldots, n$, $g_i(x, y) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^{m_1}$, is monotonic non decreasing with respect to y and monotonic non increasing with respect to x, while, for $i = 1, \ldots, n-1$, $r_i(x, y) : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^{m_2}$ is monotonic non decreasing with respect to x and monotonic non increasing with respect to y. Here, m_1, m_2 are positive natural numbers.

For $i = 2, \ldots, n$, define functions $\gamma_i : \bar{\mathbb{R}} \to \bar{\mathbb{R}}$, such that $\gamma_i(-\infty) = -\infty$ and, if $x \in \mathbb{R}$,

$$\gamma_i(x) = \begin{cases} -\infty, & \text{if } \{y \in \mathbb{R} : g_i(x, y) \le 0\} \text{ is empty} \\ \sup\{y \in \mathbb{R} : g_i(x, y) \le 0\}, & \text{otherwise.} \end{cases}$$

Similarly, for $i = 1, \ldots, n-1$, define functions $\rho_i : \bar{\mathbb{R}} \to \bar{\mathbb{R}}$, such that $\rho_i(-\infty) = -\infty$ and, if $y \in \mathbb{R}$,

$$\rho_i(y) = \begin{cases} -\infty, & \text{if} \{x \in \mathbb{R} : r_i(x, y) \le 0\} \text{ is empty} \\ \sup\{x \in \mathbb{R} : r_i(x, y) \le 0\}, & \text{otherwise.} \end{cases}$$

Note that γ_i and ρ_i are monotone non decreasing functions.

Remark 1. Problem 2 takes on the form of Problem 5 by setting $v_1^- = v_1^+ = v_s$, $v_n^- = v_n^+ = v_f$, for $i = 2, \ldots, n-1$, $v_i^- = 0$, $v_i^+ = \bar{v}_i$, $m_1 = 1$, $m_2 = 1$, $f(\mathbf{v}) = \sum_{i=1}^{n-1} h(v_i, v_{i+1})$, with

$$h(x, y) = \begin{cases} \dfrac{1}{x + y} & \text{if } x \wedge y \ge 0, x \vee y > 0 \\ +\infty & \text{otherwise,} \end{cases}$$

$$\begin{aligned} g_i(x, y) &= y^2 - x^2 - u_B, \\ \gamma_i(x) &= \sqrt{x^2 + u_B} \wedge \bar{v}_i, \ i = 2, \ldots, n, \\ r_i(x, y) &= -y^2 + x^2 + l_B, \\ \rho_i(y) &= \sqrt{y^2 - l_B} \wedge \bar{v}_i, \ i = 1, \ldots, n-1. \end{aligned}$$

Problem 5 is solved by the following Algorithm 2 which is a direct generalization of Algorithm 1.

Algorithm 2: Solution of Problem 5

input : $\mathbf{v}^-$, γ_i, $i = 2, \ldots, n$, ρ_i, $i = 1, \ldots, n-1$
output: $\mathbf{v}^*$, *Feasible*

1 $f_1 \leftarrow v_1^+$
2 **for** $i \leftarrow 2$ **to** n **do**
3 $\quad f_i = \gamma_i(f_{i-1}) \wedge v_i^+$
4 $b_n \leftarrow v_n^+$
5 **for** $i \leftarrow n-1$ **to** 1 **do**
6 $\quad b_i = \rho_i(b_{i+1}) \wedge v_i^+$
7 **for** $i \leftarrow 1$ **to** n **do**
8 $\quad v_i^* = \min\{f_i, b_i\}$
9 **if** $\mathbf{v}^* \ge \mathbf{v}^-$ **then**
10 $\quad$ *Feasible* $\leftarrow$ *True*
11 **else**
12 $\quad$ *Feasible* $\leftarrow$ *False*

The following ones are the main results of the paper.

Theorem 2. *The following statements hold:*

(i) *Problem 5 is feasible if and only if vector* $\mathbf{v}^*$*, returned by Algorithm 2, satisfies condition*

$$\mathbf{v}^* \ge \mathbf{v}^-.$$

(ii) *If Problem 5 is feasible, then vector* $\mathbf{v}^*$*, returned by Algorithm 2, is the optimal solution.*

(iii) *If Problem 5 is feasible, then vector* $\mathbf{v}^*$*, returned by Algorithm 2, is the component-wise maximum of the feasible set of Problem 5.*

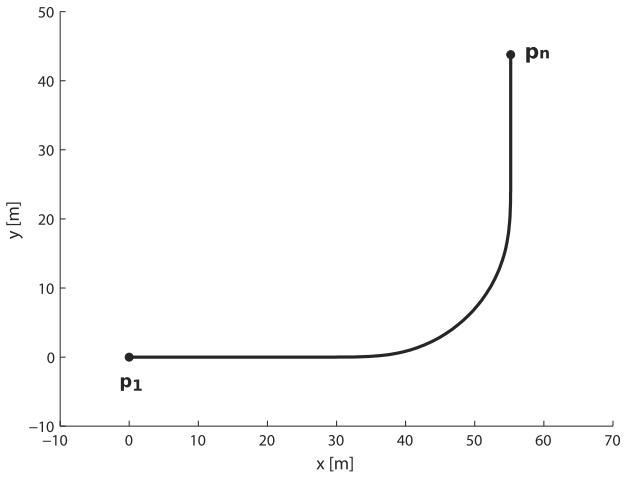

Fig. 2. Example 1: a simple G^2-path.

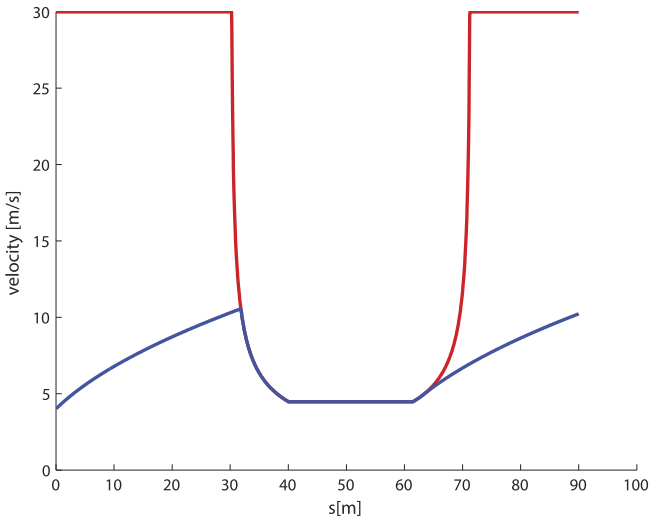

Fig. 3. Example 1 (*Forward iteration*): The red line represents the maximum velocity allowed along the path while the blue line represents the velocity sequence **f** computed after *Forward iteration*. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

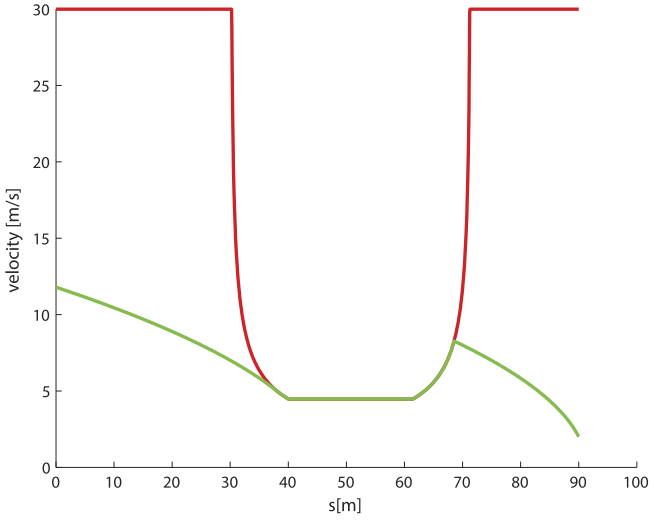

Fig. 4. Example 1 (*Backward iteration*): The red line represents the maximum velocity allowed along the path while the green line represents the velocity sequence **b** computed after *Backward iteration*. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Proofs of the results.

Theorem 2 is equivalent to Propositions 6, 7 and 3 that will be stated and proved in Section 7. Note that we presented the same results in two different ways for the sake of readability. In fact, the statement of Theorem 2 is algorithmic, while the formulations of Propositions 6, 7 and 3 require additional definitions and concepts from lattice theory that will be made in Section 7.

Theorem 3. *Algorithm 2 solves Problem 5 with linear time complexity with respect to the number of variables n. Moreover, such complexity is optimal.*

Proof. The time complexity of Algorithm 2 is linear with respect to n, since it is composed of three iterations on n. This is optimal in the sense that the time complexity of any algorithm able to solve Problem 5 is bounded from below by the $O(n)$ time required to load the problem data $\bar{v}_i$, $i = 1, \ldots, n$.

5. Examples

Example 1. As a first example consider a G^2-path composed of a line segment, a clothoid, a circle arc, a clothoid and a final line segment [6] (see Fig. 2). The minimum-time velocity planning on this path, whose total length is $s_f = 90$ m (see Problem 2), is addressed with the following data. The initial and final velocities are $v_s = 4\ \mathrm{m\,s^{-1}}$ and $v_f = 2\ \mathrm{m\,s^{-1}}$, the maximal velocity is $v_{max} = 30\ \mathrm{m\,s^{-1}}$, the longitudinal acceleration limits are $a^L_{min} = -1.5\ \mathrm{m\,s^{-2}}$ and $a^L_{max} = 1.5\ \mathrm{m\,s^{-2}}$, and the maximal normal acceleration is $a^N_{max} = 1\ \mathrm{m\,s^{-2}}$.

The number of samples is chosen as $n = 500$. Figs. 3 and 4 show the corresponding functions **f** and **b** computed as the solution of the forward iteration (12) and backward iteration (13), respectively. Fig. 5 shows the final optimal solution $\mathbf{v}^*$ computed as the minimum of **f** and **b** as defined by (14). This solution corresponds to the minimum-time $t^*_f = 16.53$ s. Note that the vehicle starts from velocity v_s and accelerates to a local maximum velocity. Then, it slows down before the beginning of the curve, in order to respect the maximum velocity constraint due to the lateral acceleration on the curve. At the end of the curve, the vehicle accelerates and reaches a second local maximum velocity after which it decelerates in order to reach the final velocity v_f.

Example 2. Consider the same path and constraints as in Example 1, with different initial and final conditions: $v_s = 4\ \mathrm{m\,s^{-1}}$, $v_f = 15\ \mathrm{m\,s^{-1}}$. Fig. 6 shows vector $\mathbf{v}^*$ obtained with Algorithm 1. In this case, Problem 2 is unfeasible by Theorem 2, being $v^*_n \neq v_f$. In fact, the allowed maximum longitudinal acceleration is not sufficient to reach the final condition on velocity.

Example 3. As a third example, consider a velocity planning on a G^2-path composed with η^2-splines [5] (see Fig. 7). A single η^2-spline is a quintic polynomial curve that can interpolate given Cartesian points with associated unit tangent vectors and curvatures. The path in Fig. 7 is composed of three η^2-splines whose interpolating data is reported in Table 1: the θ's are the angles between the x-axis and the unit tangent vectors and the k's are the curvatures. The eta parameters of these splines, which are free parameters to shape the spline path without affecting the interpolating conditions, are chosen as $\eta_1 = \eta_2 = 50$ and $\eta_3 = \eta_4 = 0$.

This planned path, which has total length $s_f = 153.05$ m, is composed of a lane-change path, an approximate clothoid, and an approximate circle arc (first, second, and third splines respectively). The velocity planning is addressed with $v_s = v_f = 0$,

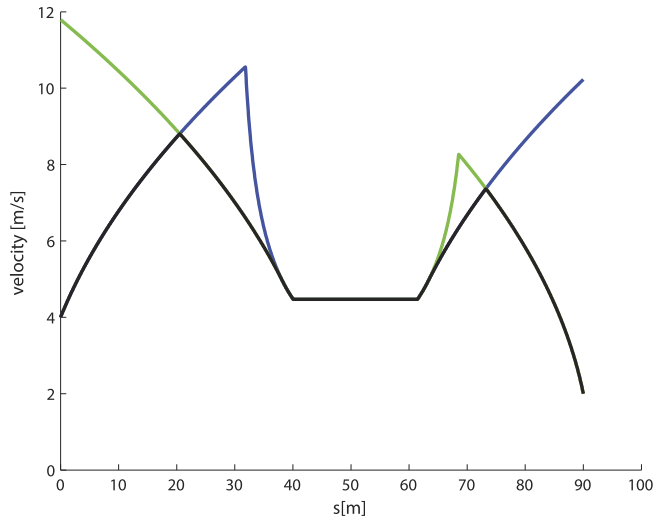

Fig. 5. Example 1 (*Join iteration*): In black the output of Algorithm 1 (the optimal velocity $\mathbf{v}^*$) is shown starting from the velocity sequences **f** and **b** previously computed.

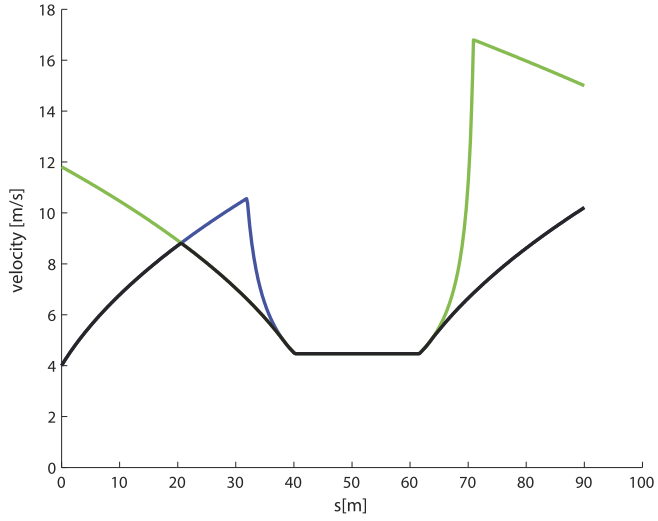

Fig. 6. Example 2 (*Join iteration*): In black the output of Algorithm 1 is shown starting from the velocity sequences **f** and **b** previously computed. The final velocity condition is not satisfied: $v_n^* \neq v_f$.

Table 1
Interpolating data of the G^2-path.

	x m	y m	θ rad	k m^{-1}
p_A	0	0	0	0
p_B	50.00	15.00	0	0
p_C	98.76	23.19	0.50	1/50
p_D	124.67	63.53	1.50	1/50

a maximal velocity $v_{\max} = 36.1\ \mathrm{m\,s^{-1}}$, longitudinal acceleration limits $a^L_{\min} = -10.5\ \mathrm{m\,s^{-2}}$, $a^L_{\max} = 4\ \mathrm{m\,s^{-2}}$, and maximal normal acceleration $a^N_{\max} = 7\ \mathrm{m\,s^{-2}}$. Algorithm 1 is applied with $n = 100$ achieving the minimum-time $t_f^* = 11.35$ s. The resulting optimal velocity profile is plotted in Fig. 8.

6. Conclusions

Velocity planning is an important issue of the automated driving of autonomous vehicles. A fast, straightforward algorithm for a velocity planning that is time-optimal has been presented. This planning takes into account the usual kinematic constraints plus a bound on the vehicle's maximum normal acceleration that helps to avoid possible dangerous sideslips.

The algorithm has $O(n)$ computational complexity and it appears easy to implement in real-time. Moreover, the algorithm's output is a velocity profile that is a function of the arc-length along the path to be followed by the automated vehicle. This feature should make the velocity profile a good reference that facilitates the tracking by the feedforward/feedback motion controller of the autonomous vehicle.

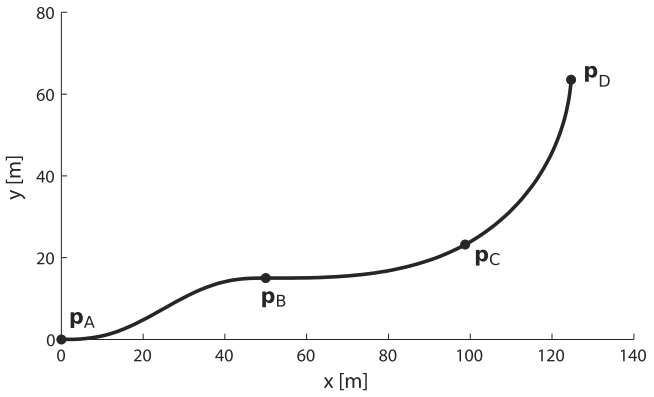

Fig. 7. Example 3: A G^2-path composed of η^2-splines [5].

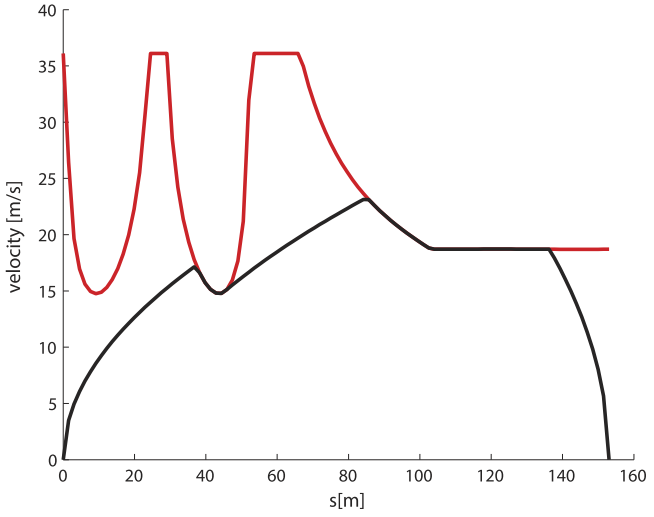

Fig. 8. Example 3: The red line represents the maximum velocity allowed along the path and the black line plots the optimal velocity profile. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

7. Proofs

Set $P = \bar{\mathbb{R}}^n$, note that $\langle P; \vee, \wedge\rangle$ is a complete lattice. Hence, for each subset $S \subseteq P$ exists the unique least upper bound $\mathbf{x} \in P$, such that:

$$(\forall \mathbf{y} \in P)[((\forall \mathbf{s} \in S)\, \mathbf{s} \leq \mathbf{y}) \iff \mathbf{x} \leq \mathbf{y}].$$

The least upper bound of S is denoted by $\bigvee S$ (see Definitions 2.1, 2.4 and Notation 2.3 on pages 33–34 of [21]).

Define functions $F, B, M : P \to P$, such that, for $\mathbf{u} \in P$, $F(\mathbf{u})$, $B(\mathbf{u})$ are given by the solutions of the difference equations

$$f_1 = u_1$$
$$f_i = \gamma_i(f_{i-1}) \wedge u_i,\ i = 2, \ldots, n,$$

$b_n = u_n$
$b_i = \rho_i(b_{i+1}) \wedge u_i, \;\; i = n-1, \ldots, 1,$

and $M(\mathbf{u}) = F(\mathbf{u}) \wedge B(\mathbf{u})$. It is apparent from the definition of M that the output $\mathbf{v}^*$ of Algorithm 2 satisfies $\mathbf{v}^* = M(\mathbf{v}^+)$.

We will refer to the following definitions.

Definition 1. A function $\phi : P \to P$ is meet-preserving if, $\forall \mathbf{x}, \mathbf{y} \in P$,

$$\phi(\mathbf{x} \wedge \mathbf{y}) = \phi(\mathbf{x}) \wedge \phi(\mathbf{y}).$$

Definition 2. A function $\phi : P \to P$ is order preserving if, $\forall \mathbf{x}, \mathbf{y} \in P$,

$$\mathbf{x} \leq \mathbf{y} \Rightarrow \phi(\mathbf{x}) \leq \phi(\mathbf{y}).$$

Proposition 1. *Functions F, B, M are meet-preserving and order preserving.*

Proof.

Let $\mathbf{u}, \mathbf{v} \in P$. Let $\mathbf{a} = F(\mathbf{u}), \mathbf{b} = F(\mathbf{v}), \mathbf{c} = F(\mathbf{u} \wedge \mathbf{v})$. We prove that $\mathbf{c} = \mathbf{a} \wedge \mathbf{b}$, by induction. First, note that $c_1 = a_1 \wedge b_1$, then, assume that $c_k = a_k \wedge b_k$, $k = 1, \ldots, i-1$. Then, since γ_i is monotonic nondecreasing, $c_i = \gamma_i(a_{i-1} \wedge b_{i-1}) \wedge u_i = (\gamma_i(a_{i-1}) \wedge u_i) \wedge (\gamma_i(b_{i-1}) \wedge u_i) = a_i \wedge b_i$. The proof that $B(\mathbf{u} \wedge \mathbf{v}) = B(\mathbf{u}) \wedge B(\mathbf{v})$ is analogous. Finally, $M(\mathbf{u} \wedge \mathbf{v}) = F(\mathbf{u} \wedge \mathbf{v}) \wedge B(\mathbf{u} \wedge \mathbf{v}) = F(\mathbf{u}) \wedge F(\mathbf{v}) \wedge B(\mathbf{u}) \wedge B(\mathbf{v}) = F(\mathbf{u}) \wedge B(\mathbf{u}) \wedge F(\mathbf{v}) \wedge B(\mathbf{v}) = M(\mathbf{u}) \wedge M(\mathbf{v})$. Since maps F, B, M are meet-preserving, they are also order preserving (see Proposition 2.19 on page 44 of [21]).

Proposition 2. *Function M satisfies the following properties, $\forall \mathbf{u} \in P$,*
(a) $M(\mathbf{u}) \leq \mathbf{u}$,
(b) $M(M(\mathbf{u})) = M(\mathbf{u})$.

Proof.

(a) It is a consequence of the definition of M.

(b) Let $\mathbf{f} = F(\mathbf{u})$, $\mathbf{b} = B(\mathbf{u})$, $\mathbf{m} = M(\mathbf{u})$, $\mathbf{a} = F(M(\mathbf{u}))$, $\mathbf{c} = B(M(\mathbf{u}))$. We prove that $\mathbf{a} = \mathbf{m}$, analogously it can be proved that $\mathbf{c} = \mathbf{m}$, which implies the thesis. Note that $a_1 = m_1$, by induction assume that $a_k = m_k$, $k = 1, \ldots, i-1$, then $a_i = \gamma_i(a_{i-1}) \wedge m_i = \gamma_i(m_{i-1}) \wedge m_i = \gamma_i(f_{i-1} \wedge b_{i-1}) \wedge (f_i \wedge b_i) = (\gamma_i(f_{i-1}) \wedge f_i) \wedge (\gamma_i(b_{i-1}) \wedge b_i) = f_i \wedge b_i = m_i$.

Proposition 3.

$$M(\mathbf{v}^+) = \bigvee \{\mathbf{x} \in P : \mathbf{x} \leq M(\mathbf{x}), \mathbf{x} \leq \mathbf{v}^+\}.$$

Proof. Set $U = \{\mathbf{x} \in P : \mathbf{x} \leq \mathbf{v}^+\}$. Note that $\langle U; \vee, \wedge \rangle$ is a sublattice of $\langle P; \vee, \wedge \rangle$, moreover, by (a) of Proposition 2, if $\mathbf{x} \in U$, then $M(\mathbf{x}) \in U$. Since M is order preserving by Proposition 1, by the Knaster–Tarski Fixpoint Theorem (Theorem 2.35 on page 50 in [21])

$$\mathbf{x}^* = \bigvee \{\mathbf{x} \in P : \mathbf{x} \leq M(\mathbf{x}), \mathbf{x} \leq \mathbf{v}^+\}$$

is such that $\mathbf{x}^* = M(\mathbf{x}^*)$, moreover $\mathbf{x}^*$ is the greatest fixed point of M such that $\mathbf{x}^* \in U$. By (b) of Proposition 2, $\mathbf{x} = M(\mathbf{v}^+)$ is also a fixed point of M, thus , by definition of $\mathbf{x}^*$, $\mathbf{x} \leq \mathbf{x}^*$. To prove that $\mathbf{x}^* = \mathbf{x}$ it remains to show that $\mathbf{x}^* \leq \mathbf{x}$. To this end assume, by contradiction, that $\mathbf{x}^* \not\leq \mathbf{x}$. Since $\mathbf{x}^* = M(\mathbf{x}^*)$, $\mathbf{x} = M(\mathbf{v}^+)$ and M is order preserving, it follows that $\mathbf{x}^* \not\leq \mathbf{v}^+$, which contradicts the definition of $\mathbf{x}^*$.

Proposition 4. *The following two statements are equivalent*
(i) Set $\{\mathbf{x} \in P : \mathbf{x} = M(\mathbf{x}), \; \mathbf{v}^- \leq \mathbf{x} \leq \mathbf{v}^+\}$ *is not empty,*
(ii) $M(\mathbf{v}^+) \geq \mathbf{v}^-$.

Proof.

(ii) $\Rightarrow$ (i) It follows from the fact that $\mathbf{x}^* = M(\mathbf{v}^+)$ is such that $M(\mathbf{x}^*) = \mathbf{x}^*$ by (b) of Proposition 2.

(i) $\Rightarrow$ (ii) By contradiction, assume that $M(\mathbf{v}^+) \not\geq \mathbf{v}^-$. Choose any $\mathbf{z} \in P$ such that $\mathbf{z} = M(\mathbf{z})$ and $\mathbf{z} \leq \mathbf{v}^+$. By Proposition 3, $M(\mathbf{z}) \leq M(\mathbf{v}^+)$, hence $\mathbf{z} \leq M(\mathbf{v}^+) \not\geq \mathbf{v}^-$ so that $\mathbf{z} \not\geq \mathbf{v}^-$. Thus, being $\mathbf{z}$ generic, set $\{\mathbf{x} \in P : \mathbf{x} = M(\mathbf{x}), \; \mathbf{v}^- \leq \mathbf{x} \leq \mathbf{v}^+\}$ is empty.

Proposition 5. *Let $\mathbf{v} \in P$, then $\mathbf{v}$ is feasible for Problem 5 if and only if $\mathbf{v}^- \leq \mathbf{v} \leq \mathbf{v}^+$ and $M(\mathbf{v}) = \mathbf{v}$.*

Proof. ($\Rightarrow$) Assume that $\mathbf{v}$ is feasible for Problem 5, we first prove that $F(\mathbf{v}) = \mathbf{v}$. Let $\mathbf{f} = F(\mathbf{v})$, note that $f_1 = v_1$, by induction, assume that $f_k = v_k$, $k = 1, \ldots, i-1$. By feasibility, $g_i(v_{i-1}, v_i) \leq 0$, so that $\gamma_i(v_{i-1}) = \max\{y \in \mathbb{R} : g_i(v_{i-1}, y) \leq 0\} \geq v_i$ and $f_i = \gamma_i(f_{i-1}) \wedge v_i = \gamma_i(v_{i-1}) \wedge v_i = v_i$. Analogously, we prove that $B(\mathbf{v}) = \mathbf{v}$. Then, by definition of M, $M(\mathbf{v}) = \mathbf{v}$. Moreover, since $\mathbf{v}$ satisfies the bounds of Problem 5, $\mathbf{v}^- \leq \mathbf{v} \leq \mathbf{v}^+$.

($\Leftarrow$) Assume $M(\mathbf{v}) = \mathbf{v}$ and set $\mathbf{f} = F(\mathbf{v})$, $\mathbf{b} = B(\mathbf{v})$. Note that $f_1 = v_1$. Moreover, for $i = 2, \ldots, n$, $f_i \geq f_i \wedge b_i = v_i$, further, $f_i = \gamma_i(f_{i-1}) \wedge v_i \leq v_i$, hence $f_i = v_i$ and $F(\mathbf{v}) = \mathbf{v}$. Analogously, $B(\mathbf{v}) = \mathbf{v}$. Then $F(\mathbf{v}) = \mathbf{v}$, implies that, for $i = 2, \ldots, n$, $v_i = \gamma_i(v_{i-1}) \wedge v_i$, so that $v_i \leq \gamma_i(v_{i-1})$ and $g_i(v_{i-1}, v_i) \leq 0$, hence constraint (17) is satisfied. Analogously, $B(\mathbf{v}) = \mathbf{v}$ implies that (18) holds.

Proposition 6. *Problem 5 is feasible if and only if $M(\mathbf{v}^+) \geq \mathbf{v}^-$.*

Proof. It is a consequence of Propositions 4 and 5.

Proposition 7. *If Problem 5 is feasible, then $\mathbf{v}^* = M(\mathbf{v}^+)$ is its optimal solution.*

Proof. By contradiction, assume that there exists a feasible $\tilde{\mathbf{v}}$ such that $f(\tilde{\mathbf{v}}) \leq f(\mathbf{v}^*)$ and, by Proposition 5, $M(\tilde{\mathbf{v}}) = \tilde{\mathbf{v}}$. Then, since f is monotonic, $\tilde{\mathbf{v}} \not\leq \mathbf{v}^*$. This is not possible since $\mathbf{v}^* = \bigvee \{\mathbf{x} \in P : \mathbf{x} \leq M(\mathbf{x}), \mathbf{x} \leq \mathbf{x}^+\} \geq \tilde{\mathbf{v}}$ by Proposition 3.

7.1. Proof of Theorem 1

We only need to prove that there always exists an optimal solution $(\mathbf{y}^*, \mathbf{v}^*)$ of Problem 4 such that all the constraints $y_i \geq \frac{1}{v_i + v_{i+1}}$ are active. The proof is done by contradiction. Let us assume that there exists an optimal solution $(\mathbf{y}^*, \mathbf{v}^*)$ of Problem 4 such that

$$y_i^* > \frac{1}{v_i^* + v_{i+1}^*}. \tag{19}$$

Then, either

$$y_i^* = \frac{v_{i+1}^* - v_i^*}{u_B}, \tag{20}$$

or (recall that $l_B < 0$)

$$y_i^* = \frac{v_i^* - v_{i+1}^*}{-l_B},$$

not both, which would be possible only if $v_i^* = v_{i+1}^*$, i.e., if $y_i^* = 0$, which cannot hold in view of (19).

Case I Let us first assume that (20) holds. Then, let us consider the following update

$$v_{i+1}' = v_{i+1}^* - \delta_{i+1},$$

where δ_{i+1} is chosen so that

$$v_{i+1}^* - \delta_{i+1} - v_i^* = \frac{u_B}{v_i^* + v_{i+1}^* - \delta_{i+1}}.$$

That also implies

$$(v_{i+1}^* - \delta_{i+1})^2 = u_B + (v_i^*)^2 \geq u_B,$$

so that

$$v^*_{i+1} > v^*_{i+1} - \delta_{i+1} \geq \sqrt{u_B}.$$

Then, we make the update

$$y'_i = \frac{v'_{i+1} - v^*_i}{u_B} = \frac{1}{v^*_i + v'_{i+1}},$$

and the constraint becomes active. Note that

$$y^*_i - y'_i = \frac{\delta_{i+1}}{u_B}. \tag{21}$$

The modification involves also the constraints associated to y_{i+1}, namely

$$\begin{aligned} u_B y_{i+1} &\geq v_{i+2} - v_{i+1} \\ -l_B y_{i+1} &\geq v_{i+1} - v_{i+2} \\ y_{i+1} &\geq \frac{1}{v_{i+1} + v_{i+2}}. \end{aligned}$$

We need to update y^*_{i+1} as follows

$$y'_{i+1} = \max\left\{ \frac{v^*_{i+2} - v^*_{i+1} + \delta_{i+1}}{u_B} \frac{v^*_{i+1} - v^*_{i+2} - \delta_{i+1}}{-l_B}, \frac{1}{v^*_{i+1} - \delta_{i+1} + v^*_{i+2}} \right\}.$$

If the maximum is equal to $\frac{v^*_{i+1} - v^*_{i+2} - \delta_{i+1}}{-l_B}$, then $y'_{i+1} \leq y^*_{i+1}$, so that the new updated solution is better than $(\mathbf{y}^*, \mathbf{v}^*)$, which is not possible in view of the optimality of the latter. If the maximum is $\frac{1}{v^*_{i+1} - \delta_{i+1} + v^*_{i+2}}$, then

$$y'_{i+1} - y^*_{i+1} \leq \frac{1}{v^*_{i+1} - \delta_{i+1} + v^*_{i+2}} - \frac{1}{v^*_{i+1} + v^*_{i+2}}$$

$$= \frac{\delta_{i+1}}{(v^*_{i+1} + v^*_{i+2})(v^*_{i+1} - \delta_{i+1} + v^*_{i+2})} \leq \frac{\delta_{i+1}}{v^*_{i+1}(v^*_{i+1} - \delta^*_{i+1})} < \frac{\delta_{i+1}}{u_B},$$

so that, once again, optimality of $(\mathbf{y}^*, \mathbf{v}^*)$ is contradicted in view of (21). If the maximum is equal to $\frac{v^*_{i+2} - v^*_{i+1} + \delta_{i+1}}{u_B}$, then $y'_{i+1} - y^*_{i+1} = \frac{\delta_{i+1}}{u_B}$, and the new solution is as good as the previous one. However, in the new solution

$$y'_{i+1} > \frac{1}{v'_{i+1} + v^*_{i+2}}.$$

Then, we can repeat the same procedure by the following update

$$v'_{i+2} = v^*_{i+2} - \delta_{i+2},$$

and, if needed, by the updates

$$v'_j = v^*_j - \delta_j, \quad j = i+3, \ldots, n-1.$$

At each update either we get to a contradiction, or we get to a new optimal solution for which the constraint

$$y_j \geq \frac{1}{v_j + v_{j+1}}$$

is active. In the latter case, once we update v^*_{n-1} into v'_{n-1}, no further update is needed and we have an optimal solution with all the above constraints active.

Case II In this case we proceed in the same way but we update v_i as follows $v'_i = v^*_i - \delta_i$, where δ_i is chosen so that

$$v^*_i - \delta_i - v^*_{i+1} = \frac{-l_B}{v^*_i - \delta_i + v^*_{i+1}}.$$

We also set

$$y'_i = \frac{v'_i - v^*_{i+1}}{-l_B},$$

so that the constraint $y_i \geq \frac{1}{v_i + v_{i+1}}$ becomes active. The rest of the proof is completely similar to Case I, the only difference being that we proceed backward from i to 2, rather than forward from $i+1$ to $n-1$.

In fact, here we considered the problem where v_1 and v_n are left free. In a more realistic situation the initial and final velocities are fixed. The result still holds if $v_1 = v_n = 0$ and, more generally, if v_1 and v_n are fixed to values such that a feasible solution of Problem 2 exists.

References

[1] ERTRAC, ERTRAC Automated Driving roadmap, July 2015, http://www.ertrac.org/index.php?page=ertrac-roadmap.

[2] A. Eskandarian (Ed.), Handbook of Intelligent Vehicles, Springer Verlag, 2012.

[3] K. Kant, S.W. Zucker, Toward efficient trajectory planning: the path–velocity decomposition, Int. J. Robot. Res. 5 (3) (1986) 72–89.

[4] T. Fraichard, T.M. Howard, Iterative motion planning and safety issue, in: A. Eskandarian (Ed.), Handbook of Intelligent Vehicles, Springer-Verlag, London, 2012, pp. 1433–1458 (Chapter 55).

[5] A. Piazzi, C. Guarino Lo Bianco, M. Bertozzi, A. Fascioli, A. Broggi, Quintic G^2-splines for the iterative steering of vision-based autonomous vehicles, IEEE Trans. Intell. Transp. Syst. 3 (1) (2002) 27–36.

[6] T. Fraichard, A. Scheuer, From Reeds and Shepp's to continuous-curvature paths, IEEE Trans. Robot. 20 (6) (2004) 1025–1035.

[7] A. Piazzi, C. Guarino Lo Bianco, M. Romano, η^3-splines for the smooth path generation of wheeled mobile robots, IEEE Trans. Robot. 23 (5) (2007) 1089–1095.

[8] F. Gravot, Y. Hirano, S. Yoshizawa, Generation of "optimal" speed profile for motion planning, in: Intelligent Robots and Systems, 2007. IROS 2007, IEEE/RSJ International Conference on, 2007, pp. 4071–4076. http://dx.doi.org/10.1109/IROS.2007.4398973.

[9] G. Lini, A. Piazzi, L. Consolini, Algebraic solution to minimum-time velocity planning, Int. J. Control Autom. Syst. 11 (4) (2013) 805–814.

[10] G. Lini, L. Consolini, A. Piazzi, Minimum-time constrained velocity planning, in: Control and Automation, 2009. MED '09. 17th Mediterranean Conference on, 2009, pp. 748–753. http://dx.doi.org/10.1109/10.1109/MED.2009.5164633.

[11] V. Muñoz, A. Ollero, M. Prado, 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, 1994, pp. 2802–2807.

[12] R. Solea, U. Nunes, Trajectory Planning with Velocity Planner for Fully-Automated Passenger Vehicles, in: IEEE Intelligent Transportation Systems Conference, ITSC '06, 2006, pp. 474–480.

[13] J. Villagra, V. Milanés, J. Pérez, J. Godoy, Smooth path and speed planning for an automated public transport vehicle, Robot. Auton. Syst. 60 (2012) 252–265.

[14] C. Chen, Y. He, C. Bu, J. Han, X. Zhang, Quartic Bézier curve based trajectory generation for autonomous vehicles with curvature and velocity constraints, in: Robotics and Automation, ICRA, 2014 IEEE International Conference on, 2014, pp. 6108–6113. http://dx.doi.org/10.1109/ICRA.2014.6907759.

[15] X. Li, Z. Sun, A. Kurt, Q. Zhu, A sampling-based local trajectory planner for autonomous driving along a reference path, in: Intelligent Vehicles Symposium Proceedings, IEEE, 2014, pp. 376–381. http://dx.doi.org/10.1109/IVS.2014.6856397.

[16] C. Guarino Lo Bianco, A. Piazzi, M. Romano, Velocity planning for autonomous vehicles, in: IEEE Intelligent Vehicles Symposium, IV2004. Parma, Italy, 2004, pp. 413–418.

[17] S. Liu, An on-line reference-trajectory generator for smooth motion of impulse-controlled industrial manipulators, in: Advanced Motion Control, 2002. 7th International Workshop on, 2002, pp. 365–370. http://dx.doi.org/10.1109/AMC.2002.1026947.

[18] L. Consolini, M. Locatelli, A. Minari, A. Piazzi, A linear-time algorithm for minimum-time velocity planning of autonomous vehicles, in: 2016 24th Mediterranean Conference on Control and Automation, MED, 2016, pp. 490–495. http://dx.doi.org/10.1109/MED.2016.7536010.

[19] R.H. Bartels, J.C. Beatty, B.A. Barsky, An Introduction to Splines for Use in Computer Graphics and Geometric Modeling, Morgan Kaufmann, Palo Alto, CA, 1995.

[20] F. Ghilardelli, G. Lini, A. Piazzi, Path generation using η^4-splines for a truck and trailer vehicle, IEEE Trans. Autom. Sci. Eng. (ISSN: 1545-5955) 11 (1) (2014) 187–203. http://dx.doi.org/10.1109/TASE.2013.2266962.

[21] B.A. Davey, H.A. Priestley, Introduction to Lattices and Order, Cambridge University Press, ISBN: 9781107717527, 2002.