2017 American Control Conference
Sheraton Seattle Hotel
May 24–26, 2017, Seattle, USA

New Method for Motion Planning for Non-holonomic Systems using Partial Differential Equations

Mohamed Ali Belabbas   Shenyu Liu

Abstract— **We present in this paper a novel approach to the long-standing problem of motion planning for non-holonomic systems. Our method is built upon a parabolic partial differential equation that arises in the study of Riemannian manifold. We show how it can be brought to bear to provide a solution to a non-holonomic motion planning problem. We illustrate the method on canonical examples, namely the unicycle, the non-holonomic integrator, and the parallel parking task for a non-holonomic car model. We also briefly address computational issues pertinent to solving this particular partial differential equation, and point out the existence of fast algorithms and the fact that the problem is easily parallelizable.**

I. Introduction

Motion planning for under-actuated systems is one of the most challenging outstanding problems at the intersection of robotics, control engineering and applied mathematics. Indeed, while motion planning for linear systems is rather straightforward in the sense that explicit solutions exist for controls driving a system from an initial state to a final state, the situation is *far* more complex in the non-linear case [1].

The difficulty of the problem originates from different issues: the non-holonomic character of the system, computational and robustness issues stemming from the nonlinearities, and the need to avoid obstacles—though this last one is not only relevant in the nonlinear case, its difficulty is vastly increased when combined with non-holonomic constraints. To address these difficulties, many ad-hoc techniques based on artificial intelligence [2] have been developed. Due to their "black-box" nature, it is difficult to assess the quality of the control obtained, or even check its validity and oftentimes these methods need to be adjusted depending on the application at hand. Other methods rely on optimal control techniques [3]. For these techniques, the issue of obstacle avoidance, because it is not a natural part of an optimal control problem, needs to be handled with particular care. The complexity of these methods is also an issue [4]. On the other end of the spectrum, methods that are on a stronger mathematical footing, such as [5], [6], [7] do not scale well to realistic scenarios involving more than a few variables. In this paper, we present a new method for motion planning. While the method is relatively simple to understand, and its implementation can be done following a straightforward procedure, its justification requires introducing notions from Riemannian geometry and partial differential equations. We address only one of these two aspects in the paper, focusing on implementation issues and analysis of the results and leaving the rigorous derivation of the result to subsequent work. Moreover, we focus here on the non-holonomic part of the motion planning problem, and leave aside the obstacle avoidance issue. We will show how obstacle avoidance can be naturally embedded in our framework in subsequent work.

Section II contains the necessary definitions and mathematical preliminaries which will be used throughout this paper. Then our new method for motion planning is presented in Section III and its application and validation on canonical examples is illustrated in Section IV. Section V concludes this paper with a discussion on the computational complexity of this method.

Mohamed Ali Belabbas and Shenyu Liu are with the Electrical and Computer Engineering department and Coordinated Science Laboratory at the University of Illinois, Urbana-Champaign. The work of M.-A. Belabbas was partially supported by the NSF grants ECCS-1351586 CAREER and ECCS-1307791.

II. Mathematical preliminaries and problem statement

We present background material needed to model the systems we will treat in this paper and our novel method for control design. The natural language for this task is the one of Riemannian geometry, but for the sake of clarity of exposition, we shall work here in the manifold $M = \mathbb{R}^n$. The methods extend naturally to the case of an arbitrary Riemannian manifold.

978-1-5090-5992-8/$31.00 ©2017 AACC

The starting point is a system linear in the control

$$\dot{x} = \sum_{i=1}^{p} u_i f_i(x) \tag{1}$$

where the f_i are differentiable **vector fields** in $\mathbb{R}^n$: $f_i : \mathbb{R}^n \to \mathbb{R}^n$. We assume that the system is under-actuated, that is $p < n$. More general situations, such as system modeled as $\dot{x} = f(x, u)$, can be treated similarly to what is done here, but at the expense of a heavier notation. We do not address the case of systems with a drift term, e.g. $\dot{x} = f_0(0) + \sum_i u_i f_i(x)$ here.

Our *objective* is the following: given two states $x_0, x_1 \in \mathbb{R}^n$, find a set of controls $u_i(t)$ that drive system (1) from x_0 to x_1 in $T > 0$ seconds. For such controls to exist, the system needs of course to be controllable—or one has to make sure that x_1 is in the reachable space from x_0. We thus start with a brief review of controllability of nonlinear systems.

A. Controllability of under-actuated systems

A fundamental operation on vector fields when studying the reachable space of a system is the **Lie bracket** of two vector fields, defined as

$$[f_i, f_j](x) = \frac{\partial f_i}{\partial x}|_x f_j(x) - \frac{\partial f_j}{\partial x}|_x f_i(x).$$

A **distribution** $\Delta(x)$ is a vector subspace of $\mathbb{R}^n$ or, in general, of $T_x M$, which depends on x. We say that a vector field f belongs to $\Delta(x)$ if $f(x) \in \Delta(x)$. Note that $f \in \Delta(x)$ does not imply that $f \in \Delta(x_1)$ for $x_1 \neq x$. Given the system (1), we define the distribution

$$\Delta_0(x) = \text{span}\{f_1(x), \dots, f_p(x)\}. \tag{2}$$

This distribution represents the space of *infinitesimal motions* the system can perform when in state x. We illustrate this on the boxcar example:

Example 1 *Consider the boxcar depicted in Fig. 1 Its dynamics is modelled according to*

$$\frac{d}{dt}\begin{pmatrix} x \\ y \\ \theta \\ \phi \end{pmatrix} = u_1 \underbrace{\begin{pmatrix} \cos\phi \\ \sin\phi \\ 0 \\ \frac{1}{d}\sin\theta \end{pmatrix}}_{f_1} + u_2 \underbrace{\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}}_{f_2}. \tag{3}$$

The distribution $\Delta_0(x)$ is given by

$$\Delta_0(x) = \text{span}\{f_1(x), f_2(x)\}.$$

When $x = (0,0,0,0)$, which corresponds to the car parked at the origin, parallel to the x-axis and with its wheel parallel to the x-axis, it is clear that the only infinitesimal motions available to the car are going forward or backwards or a rotation left/right of its front wheel. These are the two directions of $\Delta_0(0)$:

$$\Delta_0(0) = \text{span}\{f_1(0), f_2(0)\} = \text{span}\{\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}\}.$$

□

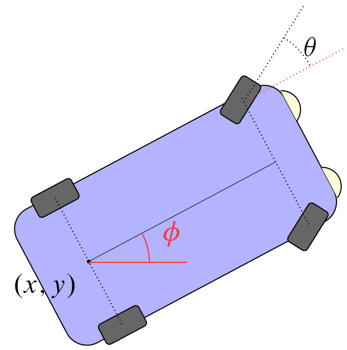


Fig. 1: A car on the plane is described by its position and orientation ($(x, y) \in \mathbb{R}^2$ and $\phi \in S^1$ respectively) and the angle of its front wheels with respect to the current orientation ($\theta \in S^1$).

A key difference between linear and nonlinear systems is that the reachable space of (1) from x_0 is the same as the reachable space of the system

$$\dot{x} = \sum_{i=1}^{p} u_i f_i(x) + \sum_{i \neq j} u_{ij} [f_i, f_j](x). \tag{4}$$

When at state x, the space of available directions of motion for system (4) is given by the distribution

$$\begin{aligned} \Delta_1(x) &= \text{span}\{f_1, \dots, f_p, [f_1, f_2], \dots, [f_{p-1}, f_p]\} \\ &= \Delta_0 \oplus \text{span}\{[f_i, f_j] \mid f_i, f_j \in \Delta_0(x)\}. \end{aligned}$$

Using this construction iteratively, we see that the distributions

$$\Delta_i(x) := \Delta_{i-1}(x) \oplus \text{span}\{[f_1, f_2](x) \mid f_1, f_2 \in \Delta_{i-1}(x)\} \tag{5}$$

are key to understanding the reachable space of Eq. (1). The precise relationship is given by Chow's theorem, which states that the system is controllable if

$$\lim_{i \to \infty} \Delta_i(x) = \mathbb{R}^n \text{ for all } x \in \mathbb{R}^n.$$

In more details

Theorem 1 (Chow) *Consider the control system* (1) *and the associated distribution* $\Delta_0(x) := \text{span}\{f_1(x), \ldots, f_p(x)\}$. *If* $\lim_{i\to\infty} \Delta_i(x) = \mathbb{R}^n$ *for all* $x \in \mathbb{R}^n$*, then the system is controllable.*

In the construction above, we added to the original system (1) *virtual controls* $u_{ij}, u_{ijk}, \ldots$ in order to study its reachable space. The corresponding distributions $\Delta_1, \Delta_2, \ldots$ do not however represent the *physical directions* of motion for the system; these are represented by the distribution Δ_0.

We call the curves in $\mathbb{R}^n$ that can be obtained by some choice of the controls $u_i(t)$ admissible curves:

Definition 1 (Admissible curves) *A curve* $v(t) : [a,b] \to \mathbb{R}^n$ *for* $a < b$ *is called admissible for system* (1) *if there exists continuous* $u_i(t) : [a,b] \to \mathbb{R}$, $i = 1, \ldots, p$ *so that*

$$\frac{d}{dt}v(t) = \sum_{i=1}^{p} u_i(t) f_i(v(t))$$

Said otherwise, a curve is admissible if it is a trajectory of the system for some choice of controls. We record here the following fact:

Lemma 1 *A curve* $v(t)$ *is admissible if and only if*

$$\frac{d}{dt}v(t) \in \Delta_0(v(t))$$

The proof is straightforward, we thus omit it. We conclude that if a system is controllable, for an arbitrary pair of $x_0, x_1 \in \mathbb{R}^n$, there exists a horizontal curve $v(t)$ joining x_0 to x_1:

Corollary 1 *Consider system* (1) *and the attached distribution* $\Delta_0(x)$ *defined in Eq.* (2). *Set* $\Delta_i(x)$ *as in Eq.* (5) . *If* $\lim_{i\to\infty} \Delta_i(x) = \mathbb{R}^n$ *for all* $x \in \mathbb{R}^n$*, then for any* $x_0, x_1 \in \mathbb{R}^n$*, there exists an admissible curve* $v(t)$ *so that* $v(0) = x_1$ *and* $v(1) = x_1$.

We conclude this section by noting that given an admissible curve joining x_0 to x_1, one can easily find controls $u_i(t)$ that drive the system from x_0 to x_1. One approach goes as follows: define the matrix

$$F_p(x) := \begin{pmatrix} | & | & & | \\ f_1(x) & f_2(x) & \cdots & f_p(x) \\ | & | & & | \end{pmatrix} \tag{6}$$

and denote by $A^\dagger$ the pseudoinverse [8] of A. Then we have

Lemma 2 *Assume that system* (1) *is controllable and let* $v(t)$ *be an admissible curve for* $\Delta_0(x)$ *so that* $v(0) = x_0$ *and* $v(1) = x_1$. *Then the controls* $u_i(t)$, $i = 1, \ldots, p$ *satisfying the equation*

$$\begin{pmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_p(t) \end{pmatrix} = F_p^\dagger(v(t))\dot{v}(t) \tag{7}$$

drive the system from x_0 *to* x_1.

We omit the proof due to space constraints, but emphasize the fact that the controls u_i need not be unique for a given $v(t)$. They are unique if and only if $\Delta_0(x)$ is of rank p everywhere.

B. Notions of Riemannian geometry

Recall that a symmetric positive definite matrix G defines an inner product in $\mathbb{R}^n$ via $\langle b_1, b_2 \rangle = b_1^\top G b_2$, for $b_1, b_2 \in \mathbb{R}^n$. Since the control systems we are considering are defined via vector fields, that is vector-valued function, it is natural to allow the inner product matrix G to similarly be a matrix-valued function $G(x)$. If $G(x)$ is symmetric, positive definite that depends smoothly on $x \in \mathbb{R}^n$, then $G(x)$ defines a **Riemannian metric** in $\mathbb{R}^n$. When the state-space is a manifold M, a Riemannian metric is defined similarly as an inner product on T_xM which depends smoothly on $x \in M$.

We now introduce the notion of length and energy of a path

Definition 2 (Length and Energy functionals) *Let* $v : [0,T] \to M$ *be a differentiable curve in a Riemannian manifold with inner product* $G(x)$. *We define the length and energy functionals as*

$$L(v) = \int_0^T \sqrt{\dot{v}^\top G \dot{v}}\,dt \text{ and } E(v) = \frac{1}{2}\int_0^T \dot{v}^\top G \dot{v}\,dt$$

respectively.

It is a well-known fact that the energy function depends on the time parametrization, but the length does not. However, their minimizers are the same [9]: if $v^*(t)$ is a minimizer of L if and only if it is a minimizer of E.

III. Methodology

We now present the new method proposed in this paper to obtain admissible paths for a system and illustrate its use on several canonical examples. We also provide a brief, intuitive justification of the method. For the sake of clarity, we will make several rather strong assumptions (such as the constant rank assumption below), but they are not required in general.

The starting point is the following: we are given the control system (1) in $\mathbb{R}^n$. We set the distribution

$$\Delta_0(x) = \operatorname{span}\{f_1(x), \dots, f_p(x)\}$$

and we assume that $\Delta_0(x)$ has rank p for all $x \in \mathbb{R}^n$. We denote by $\Delta_0^\perp(x)$ be the **orthogonal distribution** to Δ_0:

$$\Delta_0^\perp(x) := \{v \in \mathbb{R}^n \mid v^\top w = 0 \text{ for all } w \in \Delta_0(x)\}.$$

Let $f_{p+1}(x), \dots, f_n(x)$ be a basis of $\Delta_0^\perp(x)$. Hence the matrix

$$F(x) := \begin{pmatrix} | & | & & | \\ f_1(x) & f_2(x) & \cdots & f_n(x) \\ | & | & & | \end{pmatrix} \tag{8}$$

is invertible for all $x \in \mathbb{R}^n$.

Now let $c > 0$ be a positive real number and define the diagonal matrix $D_p(c)$ as the $n \times n$ matrix whose first p diagonals are 1, and next $n-p$ diagonals are c:

$$D_p(c) := \begin{pmatrix} 1 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & 0 \\ 0 & & 1 & & & \vdots \\ \vdots & & & c & & \\ & & & & \ddots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & c \end{pmatrix}. \tag{9}$$

Given $D_p(c)$ and $F(x)$ defined as above, we set $G(x)$ to be

$$G(x) := (F^\top)^{-1} D_p(c) F^{-1} \tag{10}$$

From its definition, we see that $G(x)$ is differentiable and invertible. We denote by g_{ij} the ijth entry of G and by g^{ij} the ijth entry of G^{-1}. Finally, define the functions (these are also called the Christoffels symbols of $G(x)$ [9]).

$$\Gamma^i_{jk}(x) := \frac{1}{2}\sum_l g^{il}\left(\frac{\partial g_{lj}}{\partial x_k} + \frac{\partial g_{lk}}{\partial x_j} - \frac{\partial g_{jk}}{\partial x_l}\right) \tag{11}$$

With the above data, we define a family of partial differential equations parametrized by $c > 0$. Their solutions *converge* to an *admissible* path joining x_0 and x_1 as $c \to \infty$. To this end, we denote by $v(t,s)$ a *family* of curves joining x_0 to x_1, where for each $s_0 \in [0,\infty)$ fixed, $v(t,s_0) : [0,1] \to \mathbb{R}^n$ is a curve in $\mathbb{R}^n$ with

$$v(0,s_0) = x_0 \text{ and } v(1,s_0) = x_1 \tag{12}$$

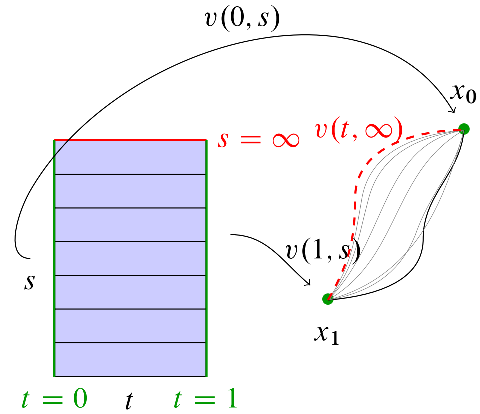


Fig. 2: For each $s \geq 0$, $v(t,s)$ is a curve joining v_0 to v_1, but not necessarily an admissible curve; $v(t,\infty)$ is admissible (dotted, red curve). We have $v(0,s) = x_0$ and $v(1,s) = x_1$ for all $s \geq 0$.

We illustrate the definition in Fig. 2. We let $v_i(t,s)$ be the ith entry of $v(t,s)$. We now define the following partial differential equation [9]:

$$\frac{\partial}{\partial s} v_i(t,s) = \frac{\partial^2}{\partial t^2} v_i(t,s) + \sum_{j,k} \Gamma^i_{jk} \frac{\partial v_j}{\partial t}\frac{\partial v_k}{\partial t} \tag{13}$$

with boundary conditions (12) and initial condition $v(t,0)$ which can be a customized initial curve. This is a system of partial differential equations, coupled by their second terms, which are *evolution equations*. That is, they can be solved using the explicit iterative procedure

$$\begin{aligned} v_i(t, s+ds) &:= \frac{\partial^2}{\partial t^2} v_i(t,s) \\ &\quad + \sum_{j,k} \Gamma^i_{jk}(v(t,s)) \frac{\partial v_j(t,s)}{\partial t}\frac{\partial v_k(t,s)}{\partial t}, \end{aligned} \tag{14}$$

where the partial derivatives are to be approximated via finite differences in the t variable.

Indeed, the curves $v(t,\cdot) : [0,1] \to \mathbb{R}^n$ act as the state-variable, and s as the time-variable for the curve evolution. For c large, we observe that the solution $v(t,\infty)$ tends to an *admissible* curve joining v_0 to v_1, that is

$$\frac{d}{dt} v(t,\infty) \in \Delta_0(v(t,\infty)).$$

We can obtain controls $u_i(t)$ from $v(t,\infty)$ as in Lemma 2.

Intuitive justification of the method: One can *intuitively understand* the method as follows. It can be shown [9] that the PDE (13) yields a curve $v(t,\infty)$ which minimizes the functional

$$E(v(\cdot,s)) := \int_0^1 v^\top(t,s)G(v(t,s))v(t,s)dt$$

For c large, one can furthermore show that this functional *penalizes* the directions in $\Delta_0^\perp(x)$. Hence, the solution $v(t,\infty)$ tends to an *admissible* curve joining v_0 to v_1, since the unadmissible directions, which are in $\Delta_0^\perp(x)$, are avoided by $v(t,\infty)$ due to their high cost.

IV. Example of applications

A. The boxcar model

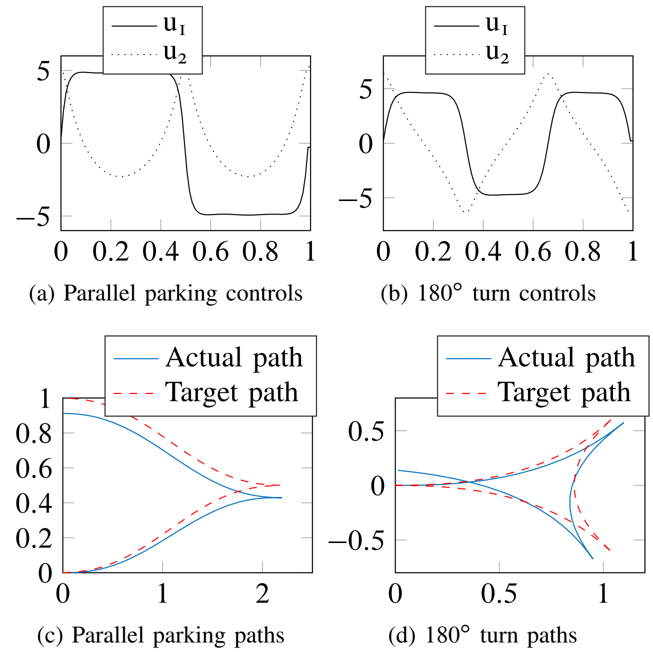
We return to the boxcar model with the dynamics (3). It is easily verified to be controllable. Denote

$$f_3 = \begin{pmatrix} -\sin\phi \\ \cos\phi \\ 0 \\ 0 \end{pmatrix}, f_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

We obtain the following entries for G:

$$\begin{aligned}
g_{11} &= \frac{c+1}{d^2}\sin^2\theta\cos^2\phi + (c-1)\sin^2\phi + 1 \\
g_{22} &= \frac{c+1}{d^2}\sin^2\theta\sin^2\phi - (c-1)\sin^2\phi + c \\
g_{33} &= 1; g_{44} = c \\
g_{12} &= g_{21} = (\frac{c+1}{d^2}\sin^2\theta - (c-1))\sin\phi\cos\phi \\
g_{14} &= g_{41} = -\frac{c}{d}\sin\theta\cos\phi \\
g_{24} &= g_{42} = -\frac{c}{d}\sin\theta\sin\phi \\
g_{13} &= g_{31} = g_{23} = g_{32} = g_{34} = g_{43} = 0
\end{aligned}$$

Notice that the physical nature of boxcar is very similar to the unicycle, and hence we expect that the "parallel parking" of a boxcar will be analogous to that of unicycle. This is indeed reflected in our simulation and hence not repeated here. However, unlike the unicycle which has 0 wheelbase and can turn without moving, boxcar has to move back and forth in order to turn around. Indeed our method illustrates an "economic" way of 180 degree turn around strategy with just 3 moves. Fig. 3 shows the obtained controls u_1 (pedal), u_2 (steering) and the comparisons between $v(t,3)$ and the solution to the system using the obtained controls for both the maneuvers of "parallel parking" and 180 degree turn around.



(a) Parallel parking controls (b) 180° turn controls

(c) Parallel parking paths (d) 180° turn paths

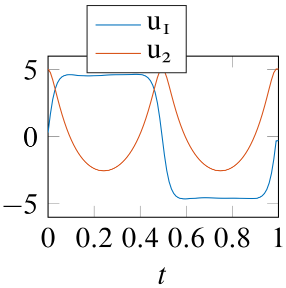
Fig. 3: 3a and 3c represent the controls u_1, u_2 and a comparison between $v(t,3)$ and the actual trajectory of the car in the xy-plane using obtained controls for $x_0 = (0,0,0,0)^\top$ and $x_1 = (0,1,0,0)^\top$ respectively. 3b and 3d represent the controls and the same comparison for moving from $(0,0,0,0)^\top$ to $(0,0,0,\pi)^\top$ respectively. The solid curves in 3c, 3d are the actual curves obtained from the controls and dashed curves are $v(t,3)$.

Remark 1 *The difference between the trajectory of the original system using the obtained control and the target path $v(t,s)$ decreases as s increases and as we increase the number of points describing a curve $v(t,\cdot)$ in our numerical implementation. In practice, different feedback control design for non-holonomic system can be implemented here for compensating this tracking error [11], [12].*

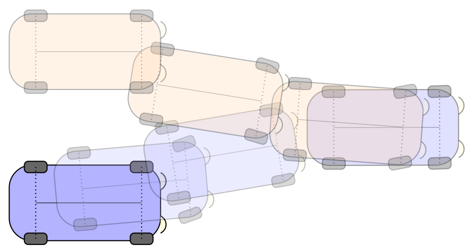
V. Complexity

At the heart of our novel method lies the *parabolic partial differential equation* (13). For the examples presented in this paper, the simulations were performed on a laptop using MATLAB and the `pdepe` function. We did not encounter any specific issues with computation speed in this case (order of seconds). The most expansive part of the method is in computing the functions Γ^i_{jk} of Eq. (11), since there are n^3 such functions. We note however that their value needs to be computed *only* at the points visited by the curve $v(t,s)$.

We have pointed earlier that Eq. (13) is an *evolution*



(a)



(b)

Fig. 4: (a). The controls u_1 and u_2 drive the car of Ex. 1 to the state $(x, y, \theta, \phi) = (0, 0, 0, 0)$ to the state $(0, 1, 0, 0)$ in one unit of time. (b). We illustrate the position of the car at times $t = 0, 0.1, 0.3, 0.5, 0.6, 0.8, 1$. A blue shading corresponds to the car moving in its forward direction, and a red shading to the car moving in reverse.

equation, and as such it can be, at least naively, solved using an Euler-type integration step $x(t + dt) = x(t) + dt\dot{x}$, but dealing with a curve $v(\cdot, s)$ in lieu of a vector $x(t)$, where s plays the role of t. In the simplest approximation, a curve is represented by a high-dimensional vector discretizing the curve; this reduces the solution to solving a high-dimensional ordinary differential equation with initial condition specified (that is the initial curve $v(t, 0)$). This is a well-studied problem with many computational implementations. We now give an estimate of the complexity of the method using a *naive* implementation. Let q be the number of discretization points of the curves $v(t, \cdot)$, and r the number of iterations in a Euler scheme solution of Eq. (14)—in our computational experiments, $r \sim 100$. Note that first and second derivatives can be obtained with 2 and 3 function evaluations respectively. The number of operations is in $O(q^3 r + rqn^3)$, where the qn^3 stems from the inversion of the matrix G at each discretization point. If this can be done symbolically once and no numerical computation is needed, then this term becomes linear in n.

Finally, we mention that one could leverage the extant literature on fast-solvers for parabolic PDEs. In particular, there is an extensive literature on parallel solvers [13], [14] for parabolic PDEs that could be brought to bear in the present case.

VI. Summary

We have presented a new method for path planning for non-holonomic systems and illustrated its used on various canonical examples, as well as validated it by numerical simulations. We did not address here the important issue of obstacle avoidance, which will be covered in a subsequent publication.

References

[1] S. M. LaValle, *Planning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2006.

[2] J.-C. Latombe, *Robot motion planning*. Springer Science & Business Media, 2012, vol. 124.

[3] J. Schultz, E. Johnson, and T. D. Murphey, *Trajectory Optimization in Discrete Mechanics*. Springer International Publishing, In Press.

[4] C. R. Hargraves and S. W. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 4, pp. 338–342, 1987.

[5] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning: Steering using sinusoids," *IEEE Transactions on Automatic Control*, vol. 38, no. 5, pp. 700–716, 1993.

[6] R. Brockett and L. Dai, "Non-holonomic kinematics and the role of elliptic functions in constructive controllability," in *Nonholonomic motion planning*. Springer, 1993, pp. 1–21.

[7] J. Laumond, *Robot motion planning and control*, ser. Lecture notes in control and information sciences. Springer, 1998.

[8] R. Horn and C. Johnson, *Matrix Analysis*. Cambridge University Press, 1990.

[9] J. Jost, *Riemannian geometry and geometric analysis*, ser. Universitext. Berlin: Springer, 1995.

[10] R. W. Brockett, "Control theory and singular riemannian geometry," in *New directions in applied mathematics*. Springer, 1982, pp. 11–27.

[11] A. De Luca, G. Oriolo, and C. Samson, *Feedback control of a nonholonomic car-like robot*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 171–253. [Online]. Available: http://dx.doi.org/10.1007/BFb0036073

[12] T. C. Lee, K. T. Song, C. H. Lee, and C. C. Teng, "Tracking control of unicycle-modeled mobile robots using a saturation feedback controller," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 305–318, 2001.

[13] G. Horton, S. Vandewalle, and P. Worley, "An algorithm with polylog parallel complexity for solving parabolic partial differential equations," *SIAM Journal on Scientific Computing*, vol. 16, no. 3, pp. 531–541, 1995.

[14] Q. Du, M. Mu, and Z. Wu, "Efficient parallel algorithms for parabolic problems," *SIAM Journal on Numerical Analysis*, vol. 39, no. 5, pp. 1469–1487, 2002.