Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online
CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-1 (2016) 601–606

Optimal Trajectory Generation for Car-type Mobile Robot using Spline Interpolation ⋆

Rahee Walambe * **Nipun Agarwal** ** **Swagatu Kale** *** **Vrunda Joshi** ****

* *PVG's College of Engineering and Technology , Pune, India(e-mail:rahee.walambe@gmail.com)*
** *PVG's College of Engineering and Technology , Pune, India*
*** *PVG's College of Engineering and Technology , Pune, India*
**** *PVG's College of Engineering and Technology , Pune, India*

Abstract: This paper focuses on the spline based trajectory generation and motion planning in a 1:10 scale down prototype of a car. The car type vehicle can be modelled as a nonholonomic system with constraints on sideways movement. The spline based trajectory generation gives us continuous, smooth and optimized path trajectories. The motion planning algorithm is based on the nonholonomic model of a car-type robot which is differentially flat in nature. Our contribution lies in the development of spline based trajectories which overcome the parametric singularities arising from the differential flatness based approach. The spline trajectories also take care of the singularities which are otherwise observed in the optimization process if the cubic polynomial based interpolation is implemented. The hardware implementation for a prototype of a car type model is also presented. Using the concept of differential flatness, we derive control inputs which steer the vehicle along this trajectory.

Keywords: Nonholonomy, differential flatness, car-type vehicle, time-reparameterization, spline polynomial

1. INTRODUCTION

In the last few decades, due to the increasing interest in mobile robots, a number of autonavigation algorithms have been developed. The autonomous vehicles are used extensively in different domains from passenger car to the hazardous applications. In order for the car type vehicle to maneuver smoothly in a given workspace, accurate planning (motion and path) algorithms are essential. There are various approaches to representing and maneuvering collision-free (admissible) configuration workspace. Latombe (1991) presents these in detail. Although these methods are suitable for holonomic tasks, they do not hold to general motion planning problem where the constraints are nonholonomic. Motion planning of nonholonomic system even in the absence of obstacles is difficult problem due to non-integrable velocity constraints. Rouchon et al. (2011), Lamiraux et al. (1999), Ramirez and Agarwal (2004) have presented an approach wherein differential flatness can be used effectively to find solution of these systems. The important aspect of differential flatness approach is that it can be used to generate control inputs without numerical approximations and integration of the system differential equations. In simple terms, flatness represents controllability in nonlinear systems as discussed in Ramirez and Agarwal (2004).

Murray et al. (1994) use the 'chain-form'transformation and find the solution to the motion planning problem by using control inputs which are integrally related sinusoidal signals. The car type system can also be shown to be nilpotent as the higher order Lie derivatives vanish and the paths can be derived using piecewise constant inputs. Path derived for given initial configuration and final configuration in Tilbury et al. (1995) are smoother due to the polynomial interpolation method adopted. Laumond et al. (1994) integrate grid based planning with nonholonomic constraints using Reed and Shepp's model discussed in Reeds and Shepp (1990). While these approaches Dubins (1961, 1957) and Reeds and Shepp (1990) give us the shortest paths joining any two configurations, the main drawback of these paths is that their curvature is not continuous. Therefore a vehicle following such a path has to stop at each discontinuity (e.g at the transition between a straight line and a circular segment), reorient it wheels, and then move forward. In Balch (1996) Dubins (1961) and Reeds and Shepp (1990), the car is considered as a point like object and the optimal path is generally the shortest path between start and end points. Since, the vehicle is assumed to be a point like object (particle), it is allowed to rotate about itself to orient along the path in the direction of the goal point. However, in a real-world car, this is not the case. That is why the purely geometric techniques developed in motion planning for holonomic systems do not apply directly to nonholonomic ones. In nonholonomic model, the constraints are applied and the smooth path is planned from start to the goal point. The resultant path is the one that an actual car can follow. The steering method gives a control algorithm which solves this path planning problem. The idea in this work is to use cubic spline based interpolation for trajectory generation. As oppose to a standard cubic polynomial interpolation, the spline interpolation is such that its point values and its first two derivatives (but not the third) are continuous at the given 'n 'points. Having zero second derivatives at the endpoints makes it natural. It is the smoothest of all possible interpolating curves in the sense that it minimizes the integral of the square of

⋆ This work is carried out under the research project grant sanctioned under the WOS-A scheme by Department of Science and Technology (DST), Govt. of India.

10.1016/j.ifacol.2016.03.121

the second derivative, thus giving a smoother and optimized trajectory.

In this work, we combine the optimization of the path with spline interpolation to generate trajectories. Initially, we designed the trajectories using cubic polynomial interpolation. The singularities which arise in the flatness based planning are handled by time-reparameterization Pedrosa et al. (2003). This does not provide solution for all cases of angular constraints. Hence we formulated SIP(Semi -Infinite) optimization problem which is used for finding the optimized (Minimal length) path. In our case, the objective being minimization of the length of the path. It is observed that this optimized cubic polynomial formulation does not generate the trajectories in certain combinations of the initial orientation θ_0 and final orientations θ_f (e.g. parallel parking scenario where the θ_0 and θ_f are same). In such cases, the singularities arise which do not give us the required trajectories. Hence we implemented the spline interpolation based trajectories which overcome this optimization singularity problem and generate smooth and continuous paths. Another contribution lies in designing of spline interpolation itself which generate the slopes at the start and end points through the optimization of the above stated SIP problem. We generate the slopes at the knot points (start and end) through the optimization of the objective function.

The paper is organized as follows: Section 2 presents the nonholonomic model of the car. Section 3 discusses differential flatness based nonholonomic motion planning for a kinematic car in detail. For this class of systems, some singularities are always associated with differential parameterization provided by flatness property. A property called 'time-reparameterization 'shown in Rouchon et al. (2011), Ramirez and Agarwal (2004) is used for handling such singularities. This is discussed in section 4 which is based on path planning between any two points using optimization and spline interpolation. The specific work in this area is highlighted in this section. Hardware implementation on 1:10 scale down model is discussed in section 5 along with results. Section 6 discusses the conclusion.

2. NONHOLONOMIC MODEL OF A CAR-TYPE ROBOT

The kinematic model of a car is derived using the assumption of pure rolling which imposes velocity level constraints which are 'non-integrable '. Such systems are classically known as nonholonomic. In addition to these nonholonomic constraints, there is a holonomic constraint due to the lower bound on the turning radius of the vehicle Laumond et al. (1994). It is important to note that the holonomic realization of a car type robot is not possible.

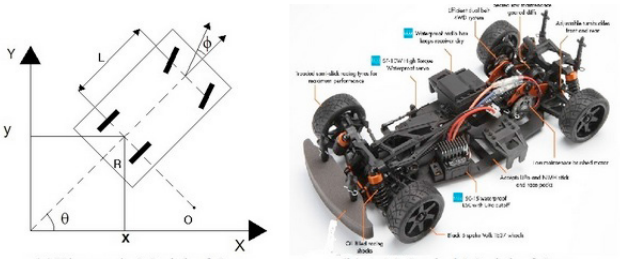
Let $\mathscr{A}$ be a rear wheel driven car-type mobile robot capable of *both* forward and reverse motion (refer Fig. 1(a)). It is modelled as a rigid rectangular body moving in a planar (two-dimensional) workspace. The rear wheels are fixed and aligned with the car while the front wheels are steerable, i.e. they are allowed to spin about the vertical axis.

It has two controls (linear and angular velocities) while it moves in a 3-dimensional configuration space given by:

$$q = (x, y, \theta) \tag{1}$$

However, control space is 2-dimensional: (u_1,u_2) with

$$u_1 = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{2}$$



(a)Kinematic Model of Car (b) 1:10 Scaled Model of Car (Swift 2 Drift Model by HPI Racing)

Fig. 1. Generalised co-ordinates of car-type robot

Where

$$\frac{\mathrm{d}x}{\mathrm{d}t} = u_1 \cos\theta \tag{3}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = u_1 \sin\theta \tag{4}$$

For a car-type robot, the nonholonomic conditions arise because of *pure rolling* and *no slipping* constraint on the wheels. Pure rolling implies that the wheels do not slip *longitudinally*, i.e along the direction of motion and no slipping implies that the wheels do not slip *transversally*. As a consequence, any path planned in the work space does not necessarily give a feasible path for the system.

By setting the lateral velocities of the rear wheels to zero and applying the rigid body constraints of the car we get the state space model of the car as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{\tan\phi}{l} \\ 0 \end{bmatrix} u_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u_2 \tag{5}$$

where u_1 and u_2 are the control inputs. Besides the nonholonomic constraints, the steering angle of the car (and hence it's turning radius) is restricted to a limiting value due to the mechanical limitation of steering mechanism of the car and is given by:

$$|\phi| \leq \phi_{max} \tag{6}$$

where ϕ_{max} is strictly positive. Fig 1(b) shows the 1:10 scaled down vehicle platform for the implementation and testing of the algorithms. The car is a Swift2drift model from HPI Racing.

3. FLATNESS BASED TRAJECTORY GENERATION

The kinematic car is a flat system where the linearizing output is the cartesian coordinates of the mid-point of the rear axle of the car. Once the linearizing output (x,y) is determined, the path planning problem becomes easy: the reference trajectory as well as the corresponding open-loop control can be expressed in terms of (x,y) and a finite number of its derivatives Rouchon et al. (2011). From the mathematical model of the car-type robot, we can determine the path the robot would trace if it were subjected to certain control inputs u_1 and u_2. However more often than not in the real world, we come across the situation where we have the path that the robot should follow (or the initial and final configurations from which we generate a path) and we have to determine the control inputs that will steer the robot along this path.

3.1 Differential Flatness

The notion of *differential flatness* was introduced by M. Fliess et al Lévine (2009). For the car-like robot, the system is flat with (*x*,*y*) as the flat outputs. The kinematic model of a car-type robot is given by Equation (5).

All the variables are time dependent. It can be seen from the model in Equation (5) that,

$$u_1 = \pm\sqrt{\dot{x}^2+\dot{y}^2} \tag{7}$$

the choice of sign depending upon whether we want forward motion (+ sign) or reverse motion (− sign).

$$\theta = \text{atan2}\,(\dot{y},\dot{x}) \tag{8}$$

which gives us the correct quadrant in which θ lies [1] .

$$\phi = \tan^{-1} l\,\frac{\dot{x}\ddot{y}-\dot{y}\ddot{x}}{(\dot{x}+\dot{y})^{\frac{3}{2}}} \tag{9}$$

$$u_2 = l\sqrt{\dot{x}^2+\dot{y}^2}\,\frac{(\dot{x}\dddot{y}-\dddot{x}\dot{y})(\dot{x}^2+\dot{y}^2)-3(\dot{x}\ddot{y}-\ddot{y}\dot{x})(\dot{x}\ddot{x}+\dot{y}\ddot{y})}{(\dot{x}^2+\dot{y}^2)^3+l^2(\dot{x}\ddot{y}-\dddot{x}\dot{y})^2} \tag{10}$$

Thus we observe from equations (7) through (10) that both the control inputs u_1 and u_2 and state variables θ and ϕ can be expressed in terms of the flat outputs and their derivatives. Therefore with the knowledge of the output trajectory of the robot, i.e. the desired path of the robot in terms of the Cartesian position of the rear wheels $(x(t),y(t))$, we can compute the control inputs required to reproduce the desired output trajectory. These depend upon the output trajectory and its third order derivatives. Hence in order to guarantee its exact reproducibility, the cartesian trajectory should be three times differentiable almost everywhere Luca et al. (1998).

4. PATH PLANNING BETWEEN TWO POINTS IN OBSTACLE-FREE SPACE

4.1 Trajectory Planning

The aim of the path planning problem is to determine a path in configuration space to move the robot from the starting position to the goal position while avoiding collisions with objects in its workspace. Based on the flat variables, x and y, given the initial and final configurations of the car, we construct trajectories $\big(x(t),y(t)\big)$ in time. We assume $x(t)$ and $y(t)$ as cubic polynomials of the order 3.

$$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{11}$$

$$\text{and} \qquad y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 \tag{12}$$

where a_0, a_1, a_2, a_3 and b_0, b_1, b_2, b_3 are constants. However the drawback of this construction is that the system is not first order controllable at the initial and final points Lévine (2009). This is because speed of the vehicle is 0 at these points, $\Rightarrow \dot{x}=0$ and $\dot{y}=0$. From Equations (8), (9) and (10) it is evident that u_2, θ, and ϕ are not defined at these points. As a consequence we are unable to account for the initial and final orientations of the car (θ_0 and θ_f) and initial and final steering angles (ϕ_0 and ϕ_f) in the construction of the trajectory. In fact for any point on the trajectory $\big(x(\tilde{t}),y(\tilde{t})\big)$ at an instant $0 \le \tilde{t} \le t_f$, the control and state trajectories are not defined when $u_1(\tilde{t}) = 0$.

[1] $\theta \in [-\pi, \pi]$

4.2 Parameterized trajectory

To avoid the case where the control and state trajectories are undefined, we use parameterized trajectory such that the path description is separated from the timing information Pedrosa et al. (2003). Let this parameter be denoted by p (for example arc length s, or a normalized variable τ) and let $p = p(t)$ be the time history along the trajectory. Therefore the desired output trajectory is expressed in terms of this path parameter as $(x(p),y(p))$ where p itself is some function of time t. Therefore now,

$$\dot{x}(t) = \frac{\mathrm{d}x(p)}{\mathrm{d}p}\cdot\frac{\mathrm{d}p}{\mathrm{d}t} \tag{13}$$

Similarly,

$$\dot{y}(t) = \frac{\mathrm{d}y(p)}{\mathrm{d}p}\cdot\frac{\mathrm{d}p}{\mathrm{d}t} \tag{14}$$

and the *pseudo-velocity* Luca et al. (1998) command is given by,

$$u_1(p) = \pm\sqrt{x'(p)^2+y'(p)^2} \tag{15}$$

where *prime* denotes differentiation with respect to path parameter. The actual linear velocity is then given as,

$$\begin{aligned} u_1(t) &= \pm\sqrt{\dot{x}(t)^2+\dot{y}(t)^2} \\ &= \pm\sqrt{x'(p)^2\,\dot{p}(t)^2+y'(t)^2\,\dot{p}(t)^2} \end{aligned} \tag{16}$$

$$u_1(t) = u_1(p)\,\frac{\mathrm{d}p}{\mathrm{d}t} \tag{17}$$

Hence, to get

$$u_1(\tilde{t}) = 0 \quad \text{for some } \tilde{t} \ge t_0$$

we set,

$$\left.\frac{\mathrm{d}p}{\mathrm{d}t}\right|_{t=\tilde{t}} = 0 \quad \text{with} \quad u_1(p) \neq 0$$

$$\Rightarrow x'(p) \neq 0 \quad ; y'(p) \neq 0 \qquad \forall p \in (0,1)$$

and hence the desired orientation, which is then computed as,

$$\theta(p) = \text{atan2}\,(y'(p),x'(p)) \tag{18}$$

is *always* well defined. Similarly zero velocity singularities are avoided in the expressions of ϕ and u_2.

Choice of parameter p We choose p to be a normalised variable expressed as a polynomial function in time. p being a normalised variable takes values in the interval [0, 1], with the mapping between p and t being such that:

$$\text{when } t = 0 \quad \rightarrow \quad p = 0 \tag{19}$$

$$\text{and when } t = t_f \quad \rightarrow \quad p = 1 \tag{20}$$

All other values $0 < t < t_f$ are mapped to corresponding values in $0 < p < 1$. Secondly there is *one-to-one* mapping between the state trajectories in p-domain and t-domain. Which means that at any time instant $t = \tilde{t}$ such that $0 \le \tilde{t} \le t_f$

$$\begin{aligned} x(\tilde{t}) &= x(p(\tilde{t})) \\ y(\tilde{t}) &= y(p(\tilde{t})) \\ \theta(\tilde{t}) &= \theta(p(\tilde{t})) \\ \phi(\tilde{t}) &= \phi(p(\tilde{t})) \end{aligned}$$

As seen in Equation (18) to set velocity to 0 we set $\dot{p} = 0$. Similarly it can be proved that to set acceleration to 0 we have to set $\ddot{p} = 0$. Thus we take p as a 5^{th} order polynomial in t –

$$p(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \alpha_3 t^3 + \alpha_4 t^4 + \alpha_5 t^5 \tag{21}$$

to satisfy the following initial and final conditions of position, velocity and acceleration –

$$p(0) = 0, p(t_f) = 1$$
$$\dot{p}(0) = 0, \dot{p}(t_f) = 0$$
$$\ddot{p}(0) = 0, \ddot{p}(t_f) = 0$$

Solving for the coefficients we get

$$p(t) = \frac{10}{t_f^3}t^3 - \frac{15}{t_f^4}t^4 + \frac{6}{t_f^5}t^5 \tag{22}$$

Parameterized polynomial Now we define the flat outputs x and y to be polynomial of cubic order of p. Therefore let,

$$x(p) = a_0 + a_1\, p + a_2\, p^2 + a_3\, p^3 \tag{23}$$
$$\text{and} y(p) = b_0 + b_1\, p + b_2\, p^2 + b_3\, p^3 \tag{24}$$

Thus subjecting them to initial conditions $(x_0, y_0, \theta_0, \phi_0)$ and final condition $(x_f, y_f, \theta_f, \phi_f)$ [2] and solving for the coefficients we get:

$$\frac{\lambda_2}{3}a_1^3 + \frac{\lambda_2(\Delta y - \lambda_3 \Delta x)}{\lambda_3 - \lambda_1}a_1^2 +$$
$$(\lambda_1 \Delta x - \Delta y)a_1 + \frac{3(\Delta y - \lambda_1 \Delta x)(\lambda_3 \Delta x - \Delta y)}{\lambda_3 - \lambda_1} = 0$$

$$a_2 = \frac{\lambda_2}{\lambda_3 - \lambda_1} \cdot a_1^2 - 2a_1 + \frac{3(\Delta x \lambda_3 - \Delta y)}{\lambda_3 - \lambda_1}$$
$$a_3 = \Delta x - a_1 - a_2$$
$$a_0 = x_0$$
$$b_3 = \Delta y - \lambda_2\, a_1^2 - \lambda_1\, a_1 - \lambda_1\, a_2$$
$$b_2 = \lambda_2\, a_1^2 + \lambda_1\, a_2$$
$$b_1 = \lambda_1\, a_1$$
$$b_0 = y_0$$

where

$$\lambda_1 = \tan\theta_0$$
$$\lambda_2 = \frac{\tan\theta_0 \sec^3\theta_0}{2l}$$
$$\lambda_3 = \tan\theta_f$$
$$\Delta x = x_f - x_0$$
$$\Delta y = y_f - y_0$$

Optimisation of trajectory We thus have a third order polynomial path $\big(x(p), y(p)\big)$ which satisfies the nonholonomic constraints of the vehicle and links the initial configuration $\big(x(0), y(0), \theta(0), \phi(0)\big)$ to the final configuration $\big(x(1), y(1), \theta(1), \phi(1)\big)$ in an obstacle free environment. If we consider any arbitrary initial configuration $\big(x(p_0), y(p_0), \theta(p_0), 0\big)$ and final configuration $\big(x(p_f), y(p_f), \theta(p_f), 0\big)$, then the feasible path $\big(x(p), y(p)\big)$ joining these two points in free space can be can be obtained by similar analysis. However it does *not* necessarily satisfy the curvature constraints, *i.e.* upper bound (ϕ_{max}) on steering angle $\phi(p), \forall p \in [0, 1]$. If the constraint on ϕ is violated, the vehicle will not be able to execute the desired trajectory when subjected to the corresponding control inputs u_1 and u_2. Also the path generated by this analysis need *not* necessarily be the path of minimum length. Hence in order to generate a path which does not violate the constraint on ϕ and is of minimal length, we modify the coefficients $a_0, \ldots, b_3$ by using an optimization routine. Our objective is to minimize the length of the path. Hence our objective function is:

$$\min \int_0^1 \sqrt{\dot{x}^2(p) + \dot{y}^2(p)}\, \mathrm{d}p \tag{25}$$

For the sake of simplicity we use the objective function Egerstedt et al. (1997):

$$\min \int_0^1 (\dot{x}^2(p) + \dot{y}^2(p))\, \mathrm{d}p \tag{26}$$

The equivalence of (25) and (26) can be proved as per Laumond et al. (1994). The constraints are the initial and final configurations and the limitation on steering angle given as $|\phi(p)| \leq \phi_{max}\ \forall p \in [0, 1]$.

Thus given the initial configuration $\big(x(0), y(0), \theta(0), \phi(0)\big)$ and final configuration $\big(x(1), y(1), \theta(1), \phi(1)\big)$, the optimization problem can be stated as:

$$\min \int_0^1 (\dot{x}^2(p) + \dot{y}^2(p))\, \mathrm{d}p \tag{27}$$

subject to the constraints

$$\begin{cases} x(0) = x_0 \\ y(0) = y_0 \\ \frac{dy}{dx}(0) = \tan\theta_0 \\ x(1) = x_f \\ y(1) = y_f \\ \frac{dy}{dx}(1) = \tan\theta_1 \end{cases} \tag{28}$$

This problem is called a *semi-infinite* programming (SIP) problem, since it is an optimization problem with finite number of variables, $a_0, a_1, a_2, a_3, b_0, b_1, b_2, b_3$ and infinite number of constraints, $|\phi(p)| \leq \phi_{max} \qquad \forall p \in [0, 1]$.

The problem can also be solved for any two arbitrary initial and final conditions $\big(x(p_0), y(p_0), \theta(p_0), \phi(p_0)\big)$ and $\big(x(p_f), y(p_f), \theta(p_f), \phi(p_f)\big)$.

Spline Interpolation The above trajectory planning with optimization does not generate solutions for certain configurations of start (θ_0) and end (θ_f) orientations, specifically in cases where the two orientations are parallel (and anti-parallel) to each other. To overcome this drawback, spline based trajectories were developed. The basic structure of the polynomial remains same as Equations (11) and (12). However, the method of calculation of coefficients changes. The data required to construct the polynomial joining the configuration $(x_0, y_0, \theta_0, 0)$ and $(x_f, y_f, \theta_f, 0)$ as the parameter p varies from 0 to 1 is $x(0)$, $x(1)$, $y(0)$, $y(1)$, $\frac{\mathrm{d}x}{\mathrm{d}p}\big|_{p=0}$, $\frac{\mathrm{d}x}{\mathrm{d}p}\big|_{p=1}$, $\frac{\mathrm{d}y}{\mathrm{d}p}\big|_{p=0}$ and $\frac{\mathrm{d}y}{\mathrm{d}p}\big|_{p=1}$

But the initial and final values of $\frac{\mathrm{d}x}{\mathrm{d}p}, \frac{\mathrm{d}y}{\mathrm{d}p}$ are not known. However we do have the value $\frac{\mathrm{d}y}{\mathrm{d}x} = \tan\theta$ at $p = 0$ and $p = 1$. Expanding by chain rule

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{\mathrm{d}y/\mathrm{d}p}{\mathrm{d}x/\mathrm{d}p} \tag{29}$$

Consider two arbitrary constants k_0 and k_f and let us re-write the above expression as

$$\left.\frac{\mathrm{d}y/\mathrm{d}p}{\mathrm{d}x/\mathrm{d}p}\right|_{p=0} = \frac{k_0 \cdot \tan\theta_0}{k_0} \tag{30}$$

Thus we take

$$\frac{\mathrm{d}x}{\mathrm{d}p}\big|_{p=0} = k_0$$

[2] assuming $\phi_0 = \phi_f = 0$ for simplicity

and

$$\frac{\mathrm{d}y}{\mathrm{d}p}\Big|_{p=0} = k_0 \cdot \tan\theta_0$$

Similarly

$$\frac{\mathrm{d}x}{\mathrm{d}p}\Big|_{p=1} = k_f \quad \text{and} \quad \frac{\mathrm{d}y}{\mathrm{d}p}\Big|_{p=1} = k_f \cdot \tan\theta_f$$

Now all the coefficients of $x(p)$ and $y(p)$ can be written in terms of k_0 and k_f. Starting with $k_0 = k_f = 1$, we solve the optimization problem minimizing the objective function in Equation (27) subject to the constraint $|\phi| < \phi_{max} \forall p \in [0,1]$. After implementation of the optimization routine we get refined values of k_0 and k_f from which we can determine the coefficients $a_0, \ldots, b_3$. The path $(x(p), y(p))$ thus obtained is of minimal length and satisfies the curvature constraints of the car-type robot(for S2D $\phi_{max} = 28^o$).

Flowchart

(1) Input start pose and goal pose from user.
(2) Define total time of travel (t_{end}).
(3) Determine $p(t)$ as in Equation (22).
(4) Define spline based trajectory as in Equations (23) and (24).
(5) Calculate the coefficients of $x(p)$ and $y(p)$ by the optimization method discussed above.
(6) Calculate $u_1(p)$ and $u_2(p)$ for all $p \in [0,1]$.
(7) Determine the control inputs as

$$u_1(t) = u_1(p) \cdot \dot{p}(t)$$

and

$$u_2(t) = u_2(p) \cdot \dot{p}(t)$$

for $t \in [0, t_{end}]$.
(8) Generate corresponding PWM signals for the driving and steering motor controllers.

5. HARDWARE IMPLEMENTATION

The car type robot platform used for the hardware realization is developed by HPI racing. The Swift2Drft(S2D) Radio Controlled vehicle is employed for the experimentation purpose. The Navstik Ivy Pro board (www.navstik.org) is used as the navigation platform. The Radio Control of S2D is eliminated and replaced by the flatness based motion planner running on navstik. It consists of a number of sensors mounted on a PCB with Cortex M4 Processor running the Pandapilot RTOS. The hardware implementation consisted of generating the control inputs to the motor drivers in the form of PWM pulses. The motion planner discussed so far generates the driving velocity u_1 and steering angle ϕ for a number of way points in the spline based trajectory. The u_1 and ϕ are represented in the PWM duty cycles and the corresponding PWM pulses are given to the driving motor driver (Nex Robotics) and steering motor driver. The vehicle steers according to this trajectory and generates the paths in real world. Fig. 2 shows the hardware setup.

Fig. 5 and Fig. 8 show the real world trajectories generated for the two test cases presented in matlab simulation results. In case I, the initial pose is : (0.5,0.3,0) and final pose is : (8.8,2.5,0). In case 2, the initial pose is : (1, 2, 0) and final pose : (8,9,-40). It can be seen that the trajectories generated in simulation and in hardware are not exactly matching. The primary reason being that the control is open loop and there is no feedback based controller implemented yet. The nonlinear feedback controller will improve the hardware results. Current

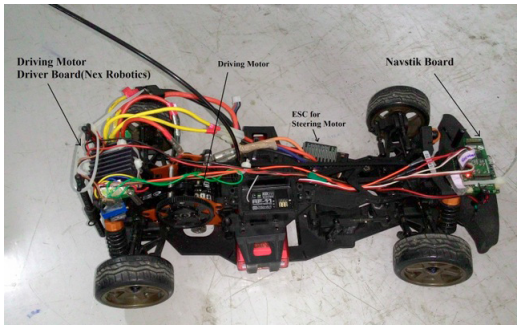


Fig. 2. Hardware Setup Consisting of Swift2Drift car model, driving motor driver and navstik board

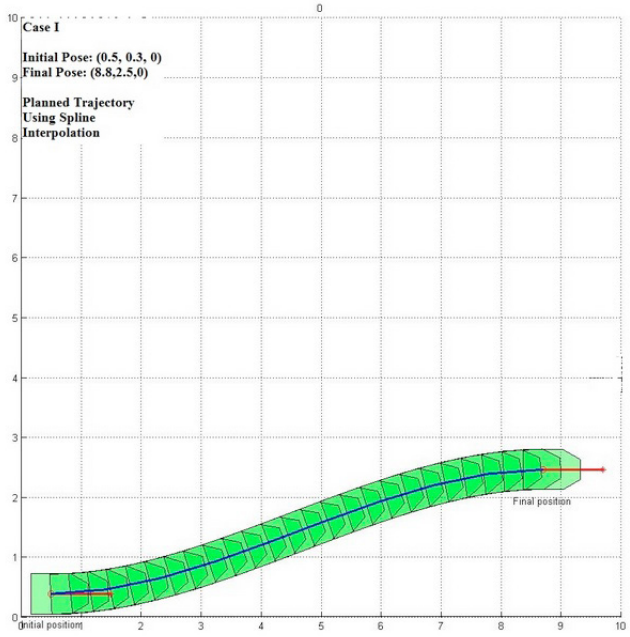


Fig. 3. Example 1: Trajectory generated by spline interpolation

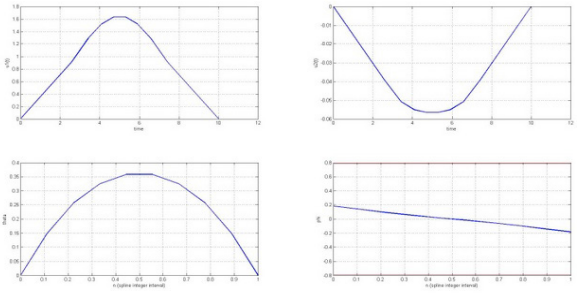


Fig. 4. Example 1: Path parameters with spline based trajectory

work is focused on the development of feedback controller design and implementation.

6. CONCLUSION

The paper presented the work carried out towards the development of an algorithm for the smooth and continuous path generation for a car type robot. Nonholonomic motion planning generates accurate and smooth trajectories whereas using differential flatness, control inputs are generated with actual

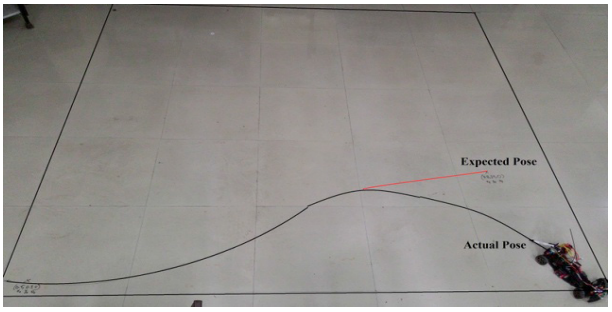


Fig. 5. Example 1: Hardware realization of trajectory on a 1:10 car prototype

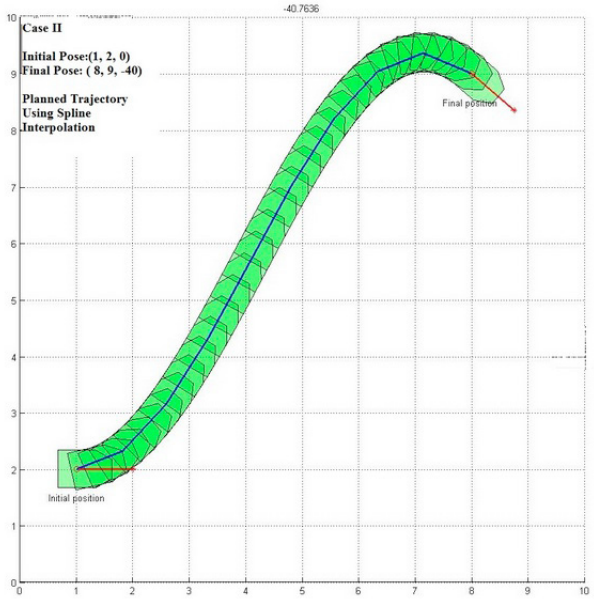


Fig. 6. Example 2: Trajectory generated by spline interpolation

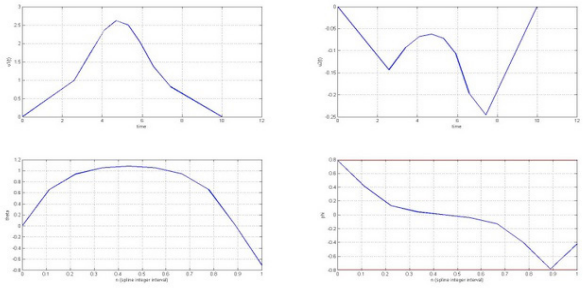


Fig. 7. Example 2: Path parameters with spline based trajectory

integration of system differential equations. To avoid the singularities observed in the cubic polynomial based trajectories, the spline based trajectories are implemented. The difference in the simulation and hardware results is due to absence of feedback controller. Current work is focused in that direction.

7. ACKNOWLEDGMENT

This work is sponsored by Department of Science and Technology (DST), Govt. of India under the research project grant sanctioned under the WOS-A scheme.

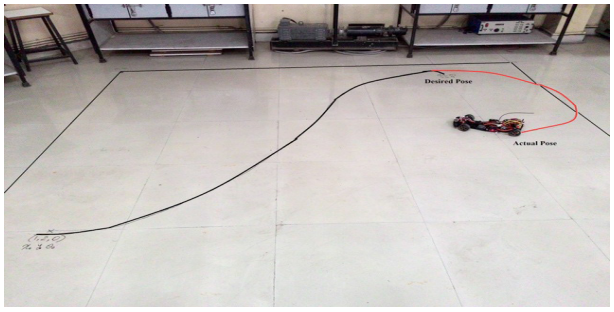


Fig. 8. Example 2: Hardware realization of trajectory on a 1:10 car prototype

REFERENCES

Balch, T. (1996). Grid-based navigation for mobile robots. *The Robotics Practitioner*, 2.

Dubins, L.E. (1957). On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, (79), 497 – 516.

Dubins, L.E. (1961). On plane curves with curvature. *Pacific Journal of Mathematics*, 11(2), 471 – 481.

Egerstedt, M., Hu, X., Rehbinder, H., and Stotsky, A. (1997). Path planning and robust tracking for a car like robot. In *Proceedings of the 5th Symposium on Intelligent Robotic Systems*.

Lamiraux, F., Sekhavat, S., , and Laumond, J.P. (1999). Motion planning and control for hilare pulling a trailer. *IEEE Transactions on Robotics and Automation*, 15(4), 640–652.

Latombe, J.C. (1991). *Robot Motion Planning*. Kluwer Academic Publishers, Boston.

Laumond, J.P., Jacobs, P.E., Taix, M., and Murray, R.M. (1994). A motion planner for nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 10(5), 577–592.

Lévine, J. (2009). *Analysis and Control of Nonlinear Systems-A Flatness-based Approach*. Springer.

Luca, A.D., Oriolo, G., and Samson, C. (1998). Feedback control of a nonholonomic car-like robot. In J.P. Laumond (ed.), *Robot Motion Planning and Control*, 171 –254. Springer.

Murray, R.M., Li, Z., and Sastry, S.S. (1994). *A Mathematical Introduction to Robotic Manipulation*. CRC Press.

Pedrosa, D.P.F., Medeiros, A.A.D., and Alsina, P. (2003). Ieee conference on robotics and automation (icra).

Ramirez, H.S. and Agarwal, S. (2004). *Differentially Flat Systems*. CRC Press.

Reeds, J.A. and Shepp, L.A. (1990). Optimal paths for a car that goes both forwards and backwards. *Pacific Journal of Mathematics*, 145(2), 367 – 393.

Rouchon, P., Fliess, M., Levine, J., and Martin, P. (2011). Flatness and motion planning : the car with n trailers. *Automatica*.

Tilbury, D., Murray, R., and Sastry, S.S. (1995). Trajectory generation for the n-trailor problem using goursat normal form. *IEEE Transactions on Automatic Control*, (5).