2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Quartic Bézier Curve based Trajectory Generation for Autonomous Vehicles with Curvature and Velocity Constraints*

Cheng Chen[1], Yuqing He[2], Chunguang Bu[2],Jianda Han[2], and Xuebo Zhang[3]

***Abstract*— To generate local trajectory between initial states and target states for autonomous vehicles, a feasible trajectory generation algorithm based on quartic Bézier curve is proposed. The problem of trajectory generation is firstly seperated into generating continuous and bounded curvature profile to shape the trajectory and generating linear velocity profile to execute the trajectory. The curvature profile generation is further converted to an optimization problem with only 3 parameters owing to the specific properties of quartic Bézier curve. Sequential quadratic programming is employed to find optimal solution with respect to specific objective function. To avoid sideslip and ensure velocity–continuity and acceleration limits, the framework of linear velocity profile generation is also proposed. A simple profile with constant acceleration is also provided as an example. Simulation results on lane keeping and changing and path following demonstrate the capability and the real–time performance of the proposed algorithm.**

I. Introduction

A. Motivation

In the last three decades, considerable efforts have been put on autonomous vehicles. As one basic ability of autonomous vehicles, trajectory generation [1] addresses the problem of determining feasible motions that drive a vehicle from initial states to target states given the associated vehicle models. This classic two–point boundary problem has been extensively studied recently. Nevertheless, it still remains unsolved considering the curvature and velocity constraints imposed by the vehicle capability.

B. Related Work

Generally speaking, trajectory generation could be categorized as: model–based algorithms and geometry–based methods.

Pioneered by Kelly and Nagy [2], most of the model–based trajectory generation algorithms follow the model predictive control (MPC) scheme. The curvature profile of the trajectory is parameterized by polynomials with respect to distance travelled. The parameters is tuned to minimize the error between target states and MPC simulated terminal states by optimization algorithms such as gradient–descent. In [3], [4], [5], the authors further extend the algorithm to various robots in different scenarios. Linear velocity profile is generated by constant, linear, linear ramp, trapezodial profile [6]. No details about the profile generation is reported. There remain three drawbacks of the algorithms in this routine. Firstly, instead of verifying feasibility during trajectory generation, the feasibility is checked after generating trajectories. Thus, the time is wasted in generating infeasible trajectories. Secondly, the optimization process is so sensitive to initial parameters that without proper initial parameters the optimization may not converge to targets. Thirdly, MPC simulations are required in each iteration of the optimization process. As a result, the trajectory generation is time–consuming. To solve the last two problems, a parameter lookup table is built to store a coarse mapping of the vehicle states and corresponding action parameters. However, hundreds of mega bytes is required to store the lookup table in the RAM. For this reason, a lot of works, such as [7],[8], have been conducted on sampling the state space.

For geometry–based trajectory generation, different geometric elements have been employed, such as line segments and arcs [9], clothoids [10], β–spline [11], Bézier curves [12], [13], [14], etc. However, only a few works have been conducted on trajectory generation while considering curvature and velocity constraints. In [11], curvature–continuous paths is generated for parallel parking maneuvers based on β–spline. The algorithm mainly focuses on generating collision–free paths while putting no constraints on curvature boundaries. Instead of normal–size vehicle with very limited steering ability, the experiments are carried only on small–size electrical vehicles. In [13], trajectory generation algorithm based on cubic Bézier curve in a multi–agent robot soccer system is proposed. Because of the special mobility of robot, the curvature constraints are not considered. Only acceleration limits are considered in the work. In [14], curvature–continuous trajectory is generated based on Bézier curves. To ensure continuous curvature while maintain numerical stability, low–degree Bézier curves are jointed to generate trajectories. The curvature, though continuous, may suffer from jerk at the conjunction points of each trajectory. No curvature boundary constraints and real–time performance are reported.

C. Contribution

In this work, the problem of trajectory generation for autonomous vehicles is studied. The proposed algorithm

*Resrach supported by the State Key Program of National Natural Science of China (Grant No. 61035005).

[1]Cheng Chen is with the State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, University of Chinese Academy of Sciences, Beijing 100049, China chenc@sia.cn

[2]Yuqing He, Chunguang Bu and Jianda Han are with the State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China heyuqing, cgbu@sia.cn, jdhan@sia.cn

[3]Xuebo Zhang is with the Institute of Robotics and Automatic Information System (IRAIS) and Tianjin Key Laboratory of Intelligent Robotics (tjKLIR), Nankai University, Tianjin 300071, China zhangxb@robot.nankai.edu.cn

proposed differs from previous work in serval ways. Firstly, the curvature of the generated trajectory is guaranteed to be continuous due to the Bézier curve itself. Secondly, optimization technique is utilized to generate bounded curvature profile considering the steering capability of the vehicles. Thirdly, the framework of generating linear velocity profile to execute the generated trajectory is proposed considering sideslip avoidance, velocity continuity and acceleration limits. A simple example is also given to demonstrate the framework. Fourthly, the proposed algorithm is insensitive to initial parameters. Thus, the algorithm could converge to the optimal solution without any lookup table.

The organization of the remaining paper is as follows. The problem of feasible trajectory generated is stated in section II. Section III introduce the preliminaries of quartic Bézier curve. The problem of trajectory generation is first seperated into generating curvature profile and linear velocity profile in section IV-A. To generate appropriate curvature profile, quartic Bézier curve is parameterized by only 3 parameters considering the initial and target states requirements in section IV-B. The curvature continuity and boundaries are discussed and solved by optimization algorithm in section IV-C and IV-D. To execute the generated trajectory, linear velocity profile is generated to avoid sideslip while satisfies the velocity–continuous and acceleration–bounded requirements in section IV-E. To demonstrate the capability of the proposed algorithm, lane changing and keeping and path following simulations are conducted in section V. Conclusion is drawn in the last section.

II. Problem Statement

Trajectory generation addresses the problem of generating actions that can drive a robot from initial states to target states. For autonomous vehicles, the states could be refined to 2–D position (x, y), heading ψ and curvature κ. As shown in Fig. 1, these states could be related by Ackerman steering mechanism and vehicle kinematic model. Let κ, v, t and

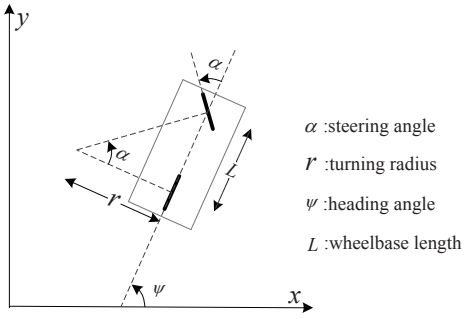

Fig. 1. Vehicle Model

s denote the curvature, linear velocity, time and distance travelled respectively. Then, the curvature, turning radius, wheelbase and steering angle could be related as [15]:

$$\kappa \triangleq \frac{1}{r} = \frac{\tan(\alpha)}{L} \tag{1}$$

As a typical nonholonomic system, the kinematics of the vehicle can be expressed in the form:

$$\begin{cases} \dot{\psi}(t) = v(t)\kappa(t) \\ \dot{x}(t) = v(t)cos(\psi(t)) \\ \dot{y}(t) = v(t)sin(\psi(t)) \end{cases} \tag{2}$$

In practical applications, the position and heading must be satisfied both in initial and desired states. However, the curvature, corresponding to steering angle, is only required in the initial states. The problem of trajectory generation could be summarized as generating actions to drive a vehicle from initial state $X_I = (x_I, y_I, \psi_I, \kappa_I)$ to target terminal state $X_T = (x_T, y_T, \psi_T)$ [1].

Because of the constraints imposed by the vehicle, trajectory generation is hardly solved yet. Typical vehicle constraints include vehicle kinematic constraints, as formulated in equation (2), and actuators limitations. Actuator limitations consist of linear velocity constraints and steering (corresponding to trajectory curvature) constraints. Therefore, three basic constraints should be satisfied to ensure the feasibility of generated trajectories for the vehicle:

- The trajectory should satisfy the vehicle *kinematic constraints*;
- The *curvature* should be *continuous* and *bounded* by the vehicle steering capability;
- The *linear velocity* to execute the generated trajectory should be *continuous, acceleration–bounded and avoiding sideslip*.

III. Quartic Bézier Curve Preliminaries

As the proposed algorithm is based on quartic Bézier curve, preliminaries on quartic Bézier curve is briefly introduced in this section. Details about Bézier curve could be found in [16]. Only the related properties are listed here. As shown in Fig. 2, a planar quartic Bézier curve with five control points satisfies the following properties:

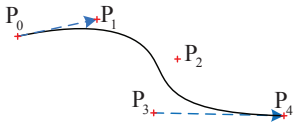

Fig. 2. Example of quartic Bézier curve

1) *Parametric Formulation*

$$P(\tau) = P_0(1-\tau)^4 + 4P_1(1-\tau)^3\tau + 6P_2(1-\tau)^2\tau^2 + 4P_3(1-\tau)\tau^3 + P_4\tau^4, \tau \in [0,1]$$

2) *Endpoint Interpolation*

$$P(0) = P_0, P(1) = P_4$$

3) *Tangent Vectors at Endpoints*

$$P'(0) = 4(P_1 - P_0), P'(1) = 4(P_4 - P_3)$$

[1]The target curvature constraint is omitted in the algorithm as the constraint greatly limits the feasible trajectories. The vehicle steering and velocity may suffers from sudden changes at junction points. Smooth techniques should be utilized to solve this problem.

4) *Curvature*

$$\kappa(\tau) = \frac{x'(\tau)y''(\tau) - y'(\tau)x''(\tau)}{(x'^2(\tau) + y'^2(\tau))^{\frac{3}{2}}}, \tau \in [0,1] \quad (3)$$

$$\kappa(0) = \frac{3}{4} \cdot \frac{|(P_1 - P_0) \times (P_2 - P_1)|}{|P_1 - P_0|^3} \quad (4)$$

5) *Invariance under Affine Transformation*

IV. Trajectory Generation Algorithm

A. Problem Simplification

To simplify the problem of trajectory generation, the velocity and steering are decoupled first. The linear velocity with respect to time and the curvature with respect to distance travelled are:

$$\begin{array}{l} v(t) = \frac{ds}{dt} \\ \kappa(s) = \frac{d\psi}{ds} \end{array} \quad (5)$$

Substitute (5) into (2) and integrate on distance travelled, the states of the vehicle could be formulated as:

$$\begin{cases} \psi(s) = \psi_0 + \int_0^s \kappa(s)ds \\ x(s) = x_0 + \int_0^s \cos(\psi(s))ds \\ y(s) = y_0 + \int_0^s \sin(\psi(s))ds \end{cases} \quad (6)$$

Obviously, the states of the vehicle and the shape of the trajectory are completely determined by the curvature on distance travelled $\kappa(s)$. Without considering vehicle drift, linear velocity only affect the time consumed to complete the generated trajectory. With this formulation, trajectory generation could be seperated into two steps: generating appropriate curvature profile that satisfies initial and target states constraints and planning linear velocity profile to execute the generated trajectory. Appropriate curvature profile should fulfill the following five requirements:

1) *The trajectory should satisfy the vehicle kinematic constraints*[2]*;*
2) *Vehicle state at $s = 0$ should be X_I;*
3) *Vehicle state at $s = s_T$ should be X_T;*
4) *To ensure continuous steering angles, $\kappa(s)$ should be continuous;*
5) *To ensure feasible steering angles, $\kappa(s)$ should be bounded by the vehicle steering capability.*

Meanwhile, the linear velocity profile should fulfill the following two requirements:

1) *Linear velocity should comply with acceleration limits;*
2) *Linear velocity should be continuous;*
3) *Linear velocity should avoid sideslip.*

B. Trajectory Parameterization

Instead of generating curvature directly, quartic Bézier curve is adopted to generate the curvature profile. Trajectory generation based on Bézier curve is not new. However, because of numerical stability, quartic and higher order Bézier curves have merely been utilized in trajectory generation. A new parameterization of quartic Bézier curve is introduced here to solve the numerical stability problem.

[2]Kinematic constraint is satisfied by the Bézier curve itself.

To simplify mathematical derivation, the states of the vehicle is firstly rotated and translated to transform X_I to $X_I = (0, 0, 0, \kappa_I)$. The trajectory will not be affected by this transformation because of the invariance of Bézier curve under affine transform. The quartic Bézier curve could be parameterized by satisfying initial and target states.

1) Requirement $X_s(0) = X_I$: The initial position of the trajectory could be met by setting $P_0 = (0, 0)$. Supposing $d_1 = |P_0P_1|$, then the initial heading could be realized by property 3) of Bézier curve:

$$P_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} d_1 \\ 0 \end{bmatrix} \quad (7)$$

Based on equation (2), the initial curvature κ_I could be satisfied if:

$$P2 = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ \frac{4\kappa_I d_1^2}{3} \end{bmatrix} \quad (8)$$

2) Requirement $X_s(s_T) = X_T$: The target position could be satisfied if $P_4 = (x_T, y_T)$. Supposing $d_4 = |P_3P_4|$, following equations should be satisfied to realize the target heading requirement:

$$P_3 = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_T - d_4\cos(\psi_T) \\ y_T - d_4\sin(\psi_T) \end{bmatrix} \quad (9)$$

To sum up, only *3 free parameters* $p = (d_1, d_4, x_2)$ are required to determine all the 5 control points of quartic Bézier curve while satisfying the initial and target state requirements.

C. Curvature Continuity and Boundaries

1) Requirement curvature continuity: With this parameterization, the initial and target states are met by default. The curvature, as formulated in equation (3), is discontinuous if and only if:

$$x'(\tau) = y'(\tau) = 0 \quad (10)$$

However, if this is the case, the curve will degenerate to a point after τ. The last control point could never be reached. That is, the curvature of Bézier curve is continuous at any point.

2) Requirement curvature boundaries: The only requirement left is the boundary requirements on curvature. The problem of curvature control of Bézier curve is almost as difficult as trajectory generation itself. Fortunately, only the maximum and minimum curvatures should be controlled to generate one feasible trajectory. The curvature represented by equation (3) could be expanded in the form:

$$\kappa(\tau) = \frac{A\tau^4 + B\tau^3 + C\tau^2 + D\tau + E}{(F\tau^6 + G\tau^5 + H\tau^4 + I\tau^3 + J\tau^2 + K\tau + L)^{\frac{3}{2}}} \quad (11)$$

Then, the curvature boundary constraint could be formulated as:

$$K_{\min} \leq \kappa(\tau) \leq K_{\max} \quad (12)$$

where $K_{\min}$ and $K_{\max}$ are the minimum and maximum feasible curvatures respectively. The curvatures are related

to steering angles by equation (1). The minimum and maximum feasible curvatures, corresponding to minimum and maximum steering angles, are decided by characteristics of the road properties and linear velocity of the vehicle, etc. Generally, the absolute values of these two boundaries should be the same. However, with impaired mobility, the absolute values of these two boundaries may be different.

D. Trajectory Optimization

In summary, the initial states, target states and curvature continuity requirements are met by the quartic Bézier curve itself. To find valid parameters that satisfying the curvature boundary requirements, optimization techniques is utilized. The objective function of the optimization could be set with respect to practical requirements. One example objective function is:

$$J(X_I, X_T, p) = \kappa_{\max}(\tau) - \kappa_{\min}(\tau), \tau \in [0, 1] \qquad (13)$$

where $\kappa_{\max}(\tau)$ and $\kappa_{\min}(\tau)$ are the maximum and minimum curvatures of $\kappa(\tau)$ respectively. The physical meaning of this objective function is that minimum steering operation, which results in smoother trajectory, is preferred. Then, the overall trajectory generation problem is finding a set of parameters minimize the objective function while satisfying the constraints. The constrained optimization problem could be formulated as:

$$\begin{aligned} \text{minimize:} \quad & J(X_I, X_T, p) \\ \text{subject to:} \quad & \kappa_{\max}(\tau) \leq K_{\max} \\ & \kappa_{\min}(\tau) \leq K_{\min} \\ & d_1 > 0 \\ & d_4 > 0 \\ & \text{etc.} \end{aligned}$$

Here in this formulation, only the curvature boundaries are constrained. Sequential quadratic programming (SQP) is employed to solve the constrained optimization problem. In each optimization iteration, the maximum and minimum curvatures are resolved by evaluating curvatures at extremes of $\kappa(\tau)$. To find the extremes, a 9–th order equation, which is obtained by $\kappa'(\tau) = 0$, is solved by numerical root finding algorithm. Then the maximum and minimum curvatures could be found by evaluating $\kappa(\tau)$ at the roots. Default initial parameters p_0 could be simply set as $(0.5, 0.5, 0.5x_T)$. As proved in the simulations, the algorithm itself is not sensitive to initial parameter.

In the trajectory generation algorithm, the curvature at target state is ignored. This is due to the fact: many applications have no requirements on target curvature; setting target curvature may greatly restrict the reachability of the target states. Nevertheless, for some applications, such as examples given in section V, the vehicle has to traverse multiple target states. If there is no restriction on target curvature, the next target may be unreachable because of the curvature of last target state. To solve this problem, desired target curvature could be set. Firstly, desired target curvature could be evaluated by circular arc or quadratic Bézier curve based on current and next target states. Then the desired target curvature with threshold is set as a constraint for the optimization process.

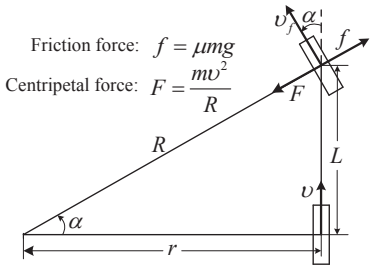

Fig. 3. Simplified vehicle model

E. Linear Velocity Profile

Until now, the trajectory that satisfies the curvature–continuity and curvature–boundary constraints is generated. To execute the generated trajectory, linear velocity should also be planned. To generate a appropriate linear velocity profile while avoiding sideslip, four steps should be take.

1) Calculate linear velocity limit to avoid sideslip. The limit could be resolved by vehicle dynamics or data from the vehicle manufacturer;
2) Appropriate linear velocity samples are selected considering the velocity limits, velocity continuity and acceleration limits;
3) Specific form, such as linear and quardratic, of the velocity profile with respect to time should be selected;
4) The parameters of the profile should be resolved by solving equations or optimization problems.

Only a simple example is introduced here to illustrate the framework. A simplified vehicle model, as shown in Fig. 3, is utilized to calculate the linear velocity limit. For the front wheel, the centripetal force should not exceeds static friction force. Let μ, g and L denote the friction coefficient, the gravitational acceleration and the wheelbase length respectively. The inequality could be formulated as:

$$\frac{mv_f^2}{R} \leq \mu mg \qquad (14)$$

Then, the linear velocity limit could be formulated as:

$$v_{\max}(\kappa) = \sqrt{\mu g(1 + L^2\kappa^2)\sqrt{\frac{1}{\kappa^2} + L^2}} \qquad (15)$$

With this formulation, the linear velocity limits could be approximated. Supposing the velocity at initial and target states are v_I and v_T respectively. These two velocities could be set considering the velocity limits. Then, one example velocity profile could be obtained by linear profile as:

$$v(t) = \frac{v_T^2 - v_I^2}{2s_T}t + v_I \qquad (16)$$

Length of the trajectory, s_T, could evaluated by integrating on the generated trajectory. The acceleration $a = (v_T^2 - v_I^2)/2s_T$ should be bounded by the acceleration limits of the vehicle. This could be achieved by setting proper

initial and target velocities. The time consumed to finish the trajectory is $2s_T/(v_T + v_I)$. More delicate profiles could be adopted considering complex vehicle dynamics and acceleration behaviors, etc. One example is shown in section V-B to demonstrate the linear velocity profile generation.

V. Simulation Results

To verify the capability of the proposed trajectory generation algorithm, simulations on lane changing and keeping, and path following are conducted. The size of the vehicle in simulation is 4.342×1.840 (L×W) m. The wheelbase length is 2.64 m. Minimum turning radius is 5.35 m. The minimum and maximum feasible curvatures are -0.187 m^{-1} and 0.187 m^{-1} respectively. The simulation is conducted on a desktop computer with Intel Core i3–2120 CPU and 4 GB RAM.

A. Lane Keeping and Changing Simulations

Lane keeping and changing are the basic requirements for vehicles travelling on the road. Three–step trajectory generation simulations is conducted to demonstrate the lane keeping and changing maneuvers. As shown in Fig. 4, initial state of the vehicle is set to be 0. Six targets, numbered as 0–6, are set for the vehicle to achieve. Corresponding curvature changes are shown in Fig. 5. The trajectory generation is divided

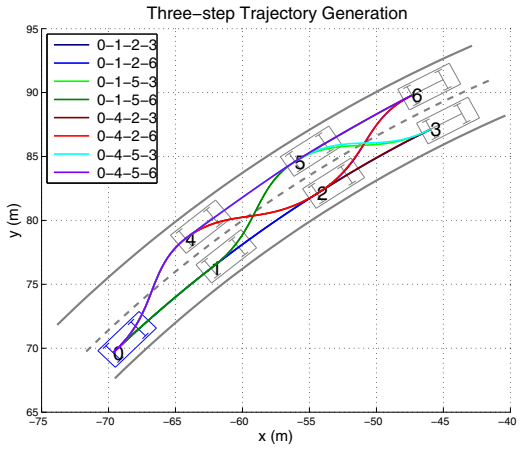

Fig. 4. Three-step lane changing and keeping

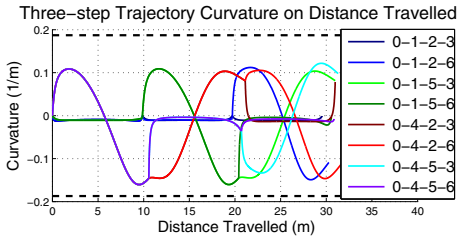

Fig. 5. Curvatures change with respect to distance travelled

into three sections. In each section, a trajectory is generated based on the algorithm proposed. The desired curvature is set as mentioned in section IV-D to ensure curvature continuous

TABLE I
Time and iteration consumed to generate trajectories

Trajectory	0-1-2-3	0-1-2-6	0-1-5-3	0-1-5-6
Time (ms)	221	790	453	507
Iterations	560	2113	1240	1370
Trajectory	0-4-2-3	0-4-2-6	0-4-5-3	0-4-5-6
Time (ms)	192	368	414	217
Iterations	504	1052	1098	597

at each intermediate target. Due to 3 lane changes, trajectory 0-4-2-6 is the most challenging trajectory. The curvature of trajectory 0-4-2-6 is still continuous and bounded. The time and iterations taken to generate all these trajectories are listed in Table I.

B. Path Following Simulations

Another typical scenario for autonomous is path following. As shown in Fig. 6, a path shaped in 'S' is generated by

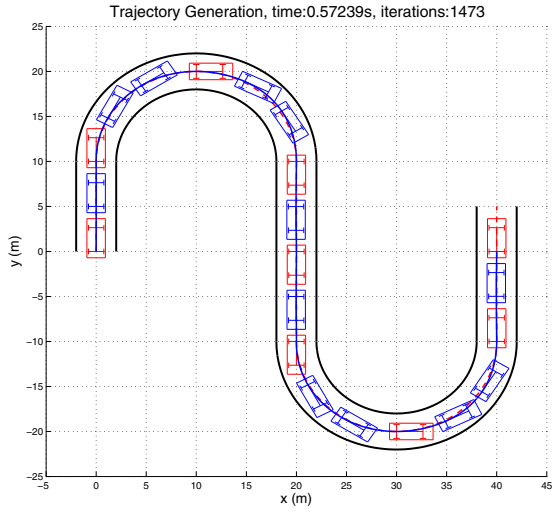

Fig. 6. Path following simulation.

connecting arcs and lines. The vehicle is commanded to follow the path from $(0, 0, \pi/2, 0)$ to $(40, 40, \pi/2)$. Six intermediate targets, the red vehicles, are specified. The vehicle trajectories are briefly represent by blue vehicles. To generated the 103 m trajectory, 8 sub–trajectory are generated. The total time and iterations consumed are 0.5724 s and 1473 respectively.

The curvature and steering angles with respect to the distance travelled are shown in Fig. 7. The dashed lines indicate the curvature limits, -0.187 and 0.187, and steering angle limits, 26.27° and -26.27°, respectively.Besides continuous and bounded, the curvature changes are minimized. The radius of the arcs is 10 m. That is, ideal curvature on these two arcs should be -0.1 and 0.1. In fact, the generated trajectory curvatures are approximately -0.1 and 0.1. The curvature only fluctuate at conjunction points.

Velocity limits and linear velocity profile are shown in Fig. 8. The maximum and minimum accelerations are set as

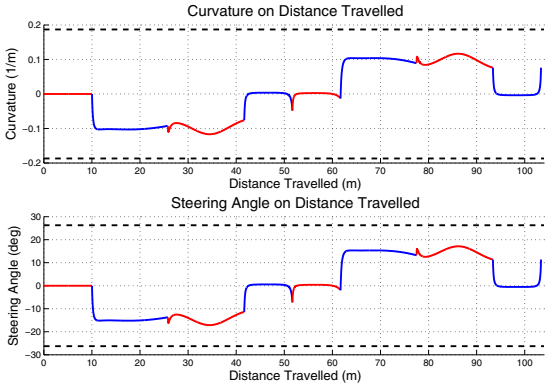

Fig. 7. Top: curvature with respect to distance travelled; Bottom: steering angles with respect to distance travelled.

$8\,m/s^2$ and $-10\,m/s^2$ respectively. The vehicle traverse most part of the trajectory at 8 m/s because of the velocity limit. It takes approximately 12 s to finish the 103 m trajectory. Velocity continuity is ensured between sections by setting the initial velocity as the terminal velocity of last section.

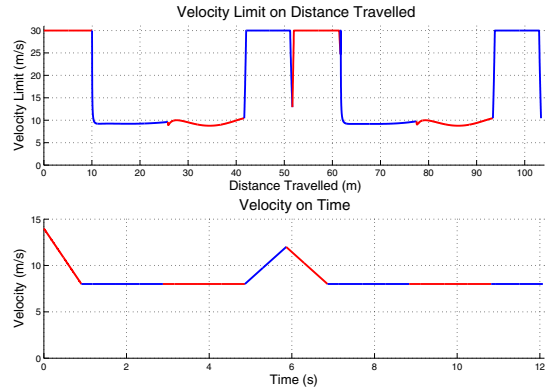

Fig. 8. Top: linear velocity limit with respect to distance travelled; Bottom: example linear velocity profile with respect to time.

Another point the authors would like to point out is the fact that the algorithm proposed is not sensitive to initial parameters. One direct proof is that in all the simulations the initial parameters are all set as default parameters.

VI. Conclusions

In this paper, the problem of generating curvature–continuous and curvature–bounded trajectories from initial states to target states for autonomous vehicles is studied. To simplify the problem, the trajectory generation is divided into generating curvature profile to shape the trajectory and generating linear velocity profile to execute the trajectory. To generate continuous and bounded curvature profile, quartic Bézier curve is applied due the specific properties. The curvature generating problem is further reduced to finding 3 suitable parameters. Optimization is adopted to find the 3 parameters with given objective function. Sequential quadratic programming is employed to find appropriate parameters. To execute the trajectory, the framework of generating continuous and acceleration–bounded linear velocity profile while considering avoiding sideslip proposed. A simple example of the frame is also introduced. To verify the capability and real–time performance, simulation on lane keeping and changing and path following are conducted. The simulation results suggest that the curvature continuity and boundaries could be ensured by the proposed algorithm. The linear velocity profile could be generated considering the velocity, acceleration and sideslip constraints. The real–time performance is also evaluated and suggest applicability to on–road autonomous vehicles. Besides, the property that the algorithm is insensitive to initial parameters is also proved.

References

[1] T. Howard and A. Kelly, "Trajectory and spline generation for all-wheel steering mobile robots," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 4827–4832.

[2] A. Kelly and B. Nagy, "Reactive nonholonomic trajectory generation via parametric optimal control," *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 583–601, 2003.

[3] T. M. Howard and A. Kelly, "Optimal rough terrain trajectory generation for wheeled mobile robots," *The International Journal of Robotics Research*, vol. 26, no. 2, pp. 141–166, 2007.

[4] D. Ferguson, T. M. Howard, and M. Likhachev, "Motion planning in urban environments," *Journal of Field Robotics*, vol. 25, no. 11-12, pp. 939–960, 2008.

[5] T. Howard, C. Green, and A. Kelly, "Receding horizon model-predictive control for mobile robot navigation of intricate paths," in *Field and Service Robotics*, ser. Springer Tracts in Advanced Robotics, A. Howard, K. Iagnemma, and A. Kelly, Eds. Springer Berlin Heidelberg, 2010, vol. 62, pp. 69–78.

[6] D. Ferguson, T. M. Howard, and M. Likhachev, "Motion planning in urban environments," *Journal of Field Robotics*, vol. 25, no. 11-12, pp. 939–960, 2008.

[7] T. M. Howard, C. J. Green, A. Kelly, and D. Ferguson, "State space sampling of feasible motions for high-performance mobile robot navigation in complex environments," *Journal of Field Robotics*, vol. 25, no. 6-7, pp. 325–345, 2008.

[8] W. Xu, J. Wei, J. Dolan, H. Zhao, and H. Zha, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 2012, pp. 2061–2067.

[9] J.-P. Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, "A motion planner for nonholonomic mobile robots," *Robotics and Automation, IEEE Transactions on*, vol. 10, no. 5, pp. 577–593, 1994.

[10] A. Scheuer and T. Fraichard, "Continuous-curvature path planning for car-like vehicles," in *Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on*, vol. 2, 1997, pp. 997–1003 vol.2.

[11] F. Gmez-Bravo, F. Cuesta, A. Ollero, and A. Viguria, "Continuous curvature path generation based on -spline curves for parking manoeuvres," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 360 – 372, 2008.

[12] Y. Li and J. Xiao, "On-line planning of nonholonomic trajectories in crowded and geometrically unknown environments," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, pp. 3230–3236.

[13] K. Jolly, R. S. Kumar, and R. Vijayakumar, "A bézier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 23 – 33, 2009.

[14] J. wung Choi, R. Curry, and G. Elkaim, "Curvature-continuous trajectory generation with corridor constraint for autonomous ground vehicles," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, 2010, pp. 7166–7171.

[15] A. Kelly and A. Stentz, "Rough terrain autonomous mobilityłpart 1: A theoretical analysis of requirements," *Autonomous Robots*, vol. 5, no. 2, pp. 129–161, 1998.

[16] M. Duncan, *Applied geometry for computer graphics and CAD*. Springer, 2005.