C2-continuous Path Planning by Combining Bernstein-Bézier Curves

Gregor Klančar, Sašo Blažič and Andrej Zdešar
Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, SI-1000, Ljubljana, Slovenia

Keywords: Path Planing, Bernstein-Bézier (BB) Curve, Splines, Optimal Velocity Profile, Wheeled Mobile Robot.

Abstract: This work proposes fifth order Bernstein-Bézier (BB) curve segments to be used in path planning approaches. The combined path consists of BB spline sections with continuous second derivative in connections which means that the path curvature is continuous and feasible for wheeled robot to drive on. To further minimize the travelling time on this path a velocity profile is optimized by considering acceleration and velocity constraints.

1 INTRODUCTION

Path planning in a known environment with obstacles presented by its map is very common task in mobile robot applications and has been widely studied in (Schwartz and Sharir, 1990), (Latombe, 1991), (LaValle, 2006). Environment usually is decomposed to cells by some algorithm like regular rectangular grid, quad trees, random sampling-based methods and the like (LaValle, 2006), (Choset et al., 2005), (Klančar et al., 2017). Among those cells an optimal collision-free path need to be find connecting current robot position and the goal location. The most commonly used is A star algorithm which returns optimal sequence of connected straight lines through the cells centers towards the goal location. Such combined path does not have continuous first and second derivative (is not C1 and C2 continuous). C1 not continuous path means that the robot following this path would have step changes of orientation while C2 discontinuous means that the robot angular velocity or also path curvature κ has step changes. Therefore the calculated path need to be smoothed to become feasible for the robot to follow it. The first studies to obtain the shortest smooth paths consisting of straight lines and circular arc was performed by (Dubins, 1957). His paths are only C1 continuous as they have discontinuous curvature.

Some possible smoothing approaches are as follows. A funnel algorithm is proposed in (Kallmann, 2005) to further optimize the path inside the corridor defined by the cells contained on the optimal path. For path optimization and smoothing inside the corridor a Fast marching method (Sethian, 1999) can be applied or smooth path generation using B-splines as in (Berglund et al., 2011). Several path smoothing ideas using local nonlinear optimization and non-parametric optimization using conjugated-gradient solution are described in (Dolgov et al., 2008). Often sharp transitions on the path e.g. corners are smoothed by inserting smooth parametric curves such as circular arcs (Yang and Wushan, 2015), Bezier curves (Choi et al., 2010), clothoids or higher order polynomials (Brezak and Petrović, 2014), (Sencer et al., 2015) enabling C2 continuous transitions.

Path smoothing is often not integrated in path planning but is usually done after the optimal path is found. This however requires additional collision checks and can influence path optimality. Several local path planners ware proposed to find smooth path sections between initial and target pose in obstacles free space as in (Chen et al., 2014) where four order Bézier curves are applied to obtain continuous and bounded curvature path. However finding collision safe, smooth and optimal path in complex environments with obstacles remains a challenging task. To cope with it a hybrid A star algorithm (Dolgov et al., 2008) was proposed which can find drivable reference path for wheeled robots. The path usually consists of curves obtained by setting constant robot commands.

This work addresses continuous path planing problem where we suggest the path to be composed of Bernstein-Bézier curves with continuous velocity and curvature transitions. The obtained path can therefore be directly driven by wheeled mobile robots. To achieve the shortest travelling time a driving velocity optimization approach is performed by considering robot capabilities. The main idea is to drive with maximal allowed accelerations and velocity to avoid

Klančar, G., Blažič, S. and Zdešar, A.
C2-continuous Path Planning by Combining Bernstein-Bézier Curves.
DOI: 10.5220/0006406602540261
In *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2017) - Volume 2*, pages 254-261
ISBN: Not Available

wheel slipping. Several path and velocity planing examples are illustrated.

Main paper contributions are two. The first is the definition of fifth order Bézier curve sections which can easily be applied to compose a C2 continuous path in some path planning applications. The second contribution is optimal velocity profile calculation approach for a spline path consisting of more Bézier curves.

2 PATH COMPOSITION IN CONTINUOUS PATH PLANING

Resulting path in most path planing approaches is composed of path sections which are continuously joined. Usually the search is done in discrete space by discretization of all possible robot poses (e.g. grid-based presentation of environment) to a finite set. Other very often used approach is to discretize input commands while the pose remains continuously defined as it is usually done in continuous path planing approaches (e.g. hybrid A star). The former can be applied to differential drive robot which commands are linear velocity $v(t)$ and angular velocity $\omega(t)$. In each node (robot pose) the path planing algorithm expands the search in a predefined number of travelling curves obtained by setting some constant translational velocity $v(t) = v_{CONST}$ and angular velocity $\omega(t) \in [\omega_{MIN}, \cdots, 0, \cdots, \omega_{MAX}]$. The path sections therefore have circular shape and the final robot pose $(x(t_F), y(t_F), \varphi(t_F))$ at time $t_F = t_S + \Delta t$ is obtained by integration

$$\begin{aligned} x(t_F) &= \int_{t_S}^{t_F} v(t)\cos(\varphi(t))dt + x(t_S) \\ y(t_F) &= \int_{t_S}^{t_F} v(t)\sin(\varphi(t))dt + y(t_S) \\ \varphi(t_F) &= \int_{t_S}^{t_F} \omega(t)dt + \varphi(t_S) \end{aligned} \tag{1}$$

which exact solution is

$$\begin{aligned} x(t_F) &= x(t_S) + \tfrac{\Delta s}{\Delta\varphi}\left(\sin(\varphi(t_S)+\Delta\varphi) - \sin(\varphi(t_S))\right) \\ y(t_F) &= y(t_S) - \tfrac{\Delta s}{\Delta\varphi}\left(\cos(\varphi(t_S)+\Delta\varphi) - \cos(\varphi(t_S))\right) \\ \varphi(t_F) &= \varphi(t_S) + \Delta\varphi \end{aligned} \tag{2}$$

where t_S is starting time, t_F is final time, Δt time increment for the path section, $\Delta s = v\Delta t$ is travelled distance and $\Delta\varphi = \omega\Delta t$ change of robot orientation.

An example of search expansion using circular paths (e.g. as in hybrid A star) expansion tree (where $x(0) = y(0) = 0$, $\varphi(0) = \pi/4$, $v = 0.5$, $\omega \in [-1, -0.5, 0, 0.5, 1]$ and $\Delta t = 1$,) is given in Fig. 1.

To follow the thick path in Fig. 1 robot controls need to be as shown in Fig. 2 which obviously is not C2 continuous because $\omega(t)$ is discontinuous.

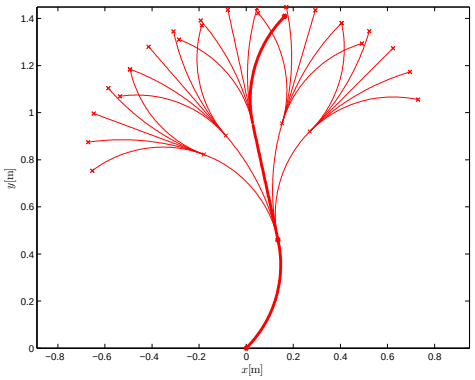

Figure 1: Search expansion using circular paths.

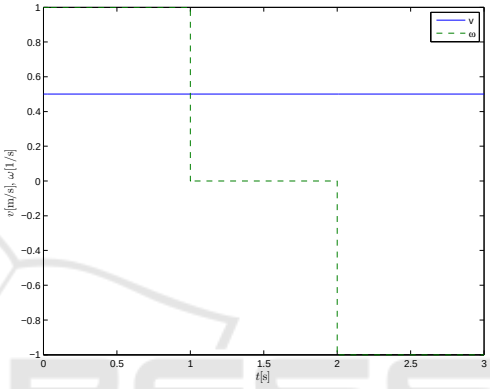

Figure 2: Differential drive robot control signal to follow thick path in Fig. 1.

To have feasible planed path for the robot a C2 continuous Bernstein-Bézier (BB) curves are proposed as follows. To achieve C2 continuous path a proper spline of two connecting BB curves need to be achieved.

To achieve this at least forth order BB curve $\mathbf{r}(\lambda)$ need to be selected. It is defined by five control points $\mathbf{P}_i = [x_i, y_i]^T$, $i \in = 0, 1, \cdots, 4$ as follows

$$\begin{aligned} \mathbf{r}(\lambda) &= (1-\lambda)^4\mathbf{P}_0 + 4\lambda(1-\lambda)^3\mathbf{P}_1 \\ &+ 6\lambda^2(1-\lambda)^2\mathbf{P}_2 + 4\lambda^3(1-\lambda)\mathbf{P}_3 + \lambda^4\mathbf{P}_4 \end{aligned} \tag{3}$$

where λ is a normalized time ($0 \le \lambda \le 1$). In this section without loss of generality assume $\lambda = t$. A C2 spline of two BB curves $\mathbf{r}_j$ and $\mathbf{r}_{j+1}$ is obtained by setting the following conditions

$$\begin{aligned} \lim_{\lambda\to 1}\mathbf{r}_j(\lambda) &= \lim_{\lambda\to 0}\mathbf{r}_{j+1}(\lambda) \\ \lim_{\lambda\to 1}\frac{d\mathbf{r}_j(\lambda)}{d\lambda} &= \lim_{\lambda\to 0}\frac{d\mathbf{r}_{j+1}(\lambda)}{d\lambda} \\ \lim_{\lambda\to 1}\frac{d^2\mathbf{r}_j(\lambda)}{d^2\lambda} &= \lim_{\lambda\to 0}\frac{d^2\mathbf{r}_{j+1}(\lambda)}{d^2\lambda} \end{aligned} \tag{4}$$

saying that the end of the curve j and the start of the curve $j+1$ as well as their first and second derivative need to coincide. From (4) the conditions for selection of the $j+1$ BB curve control points $\mathbf{P}_{i,j+1}$ re-

lated to control points of j-th curve ($\mathbf{P}_{i,j}$) selection reads

$$\begin{aligned}\mathbf{P}_{0,j+1} &= \mathbf{P}_{4,j}\\ \mathbf{P}_{1,j+1} &= 2\mathbf{P}_{4,j} - \mathbf{P}_{3,j}\\ \mathbf{P}_{2,j+1} &= 4\mathbf{P}_{4,j} - 4\mathbf{P}_{3,j} + \mathbf{P}_{2,j}\end{aligned} \tag{5}$$

To have similar spread of paths sections as in Fig. 1 the last control point $\mathbf{P}_{4,j+1}$ of BB curves is calculated using final position $x(t_F)$, $y(t_F)$ calculated by (2). While final curve orientation is achieved by setting $\mathbf{P}_{3,j+1}$ according to the final orientation $\varphi(t)$

$$\mathbf{P}_{3,j+1} = \mathbf{P}_{4,j+1} + \frac{1}{4}v\begin{bmatrix}\cos(\varphi(t_F)+\pi)\\ \sin(\varphi(t_F)+\pi)\end{bmatrix} \tag{6}$$

Path expansion during path-planing using BB curves is shown in Fig. 3. Initial points of the first BB curve are

$$\begin{aligned}\mathbf{P}_{0,j=1} &= [x(0), y(0)]^T\\ \mathbf{P}_{1,j=1} &= \mathbf{P}_{0,j=1} + 0.25v[\cos\varphi(0), \sin\varphi(0)]^T\\ \mathbf{P}_{2,j=1} &= 0.5\mathbf{P}_{1,j=1} + 0.5\mathbf{P}_{3,j=1}\end{aligned}$$

while $\mathbf{P}_{3,j=1}$ and $\mathbf{P}_{4,j=1}$ are defined considering next curve segments and relations (5). The obtained graph tree of paths has the same spread as in Fig. 1 but with continuous second derivative as seen in Figs. 3 and 4.

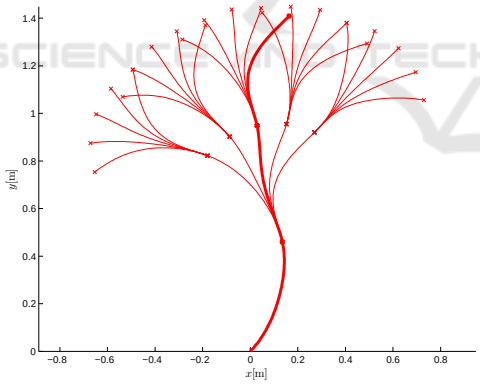

Figure 3: Search expansion using C2 continuous BB paths.

However having a closer look of Figs. 3 and 4 one can observe some unnecessary winding of the resulting paths. This is easily seen in the second path section which start and end direction are the same but the path between the start and end point is not straight as it could be. To improve this behaviour BB curves of the fifth order are used

$$\begin{aligned}\mathbf{r}(\lambda) &= (1-\lambda)^5\mathbf{P}_0 + 5\lambda(1-\lambda)^4\mathbf{P}_1\\ &+10\lambda^2(1-\lambda)^3\mathbf{P}_2 + 10\lambda^3(1-\lambda)^2\mathbf{P}_3\\ &+5\lambda^4(1-\lambda)\mathbf{P}_4 + \lambda^5\mathbf{P}_5\end{aligned} \tag{7}$$

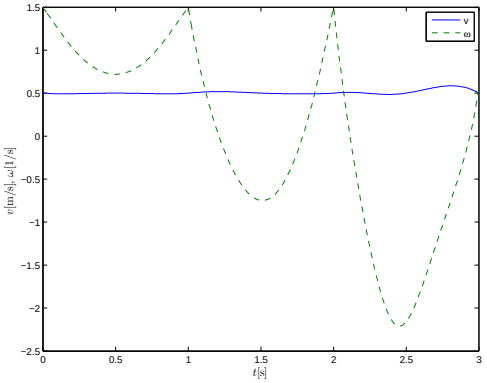

Figure 4: Differential drive robot control signal to follow thick path in Fig. 3. Angular velocity is continuous while translational velocity is similar to that in Fig. 2.

and additional conditions besides those in (4) is defined to obtain zero angular velocity and zero tangential acceleration at the curve end. This is defined as follows $\lim_{\lambda\to 1}\frac{d\varphi_{j+1}(\lambda)}{d\lambda} = 0$ and $\lim_{\lambda\to 1} a_{t,j+1} = 0$. The resulting control points reads

$$\begin{aligned}\mathbf{P}_{0,j+1} &= \mathbf{P}_{5,j}\\ \mathbf{P}_{1,j+1} &= 2\mathbf{P}_{5,j} - \mathbf{P}_{4,j}\\ \mathbf{P}_{2,j+1} &= 4\mathbf{P}_{5,j} - 4\mathbf{P}_{4,j} + \mathbf{P}_{3,j}\\ \mathbf{P}_{3,j+1} &= 2\mathbf{F} - \mathbf{E}\\ \mathbf{P}_{4,j+1} &= \mathbf{F}\\ \mathbf{P}_{5,j+1} &= \mathbf{E}\end{aligned} \tag{8}$$

where $\mathbf{E} = [x(t), y(t)]^T$, $\mathbf{F} = \mathbf{E} + 0.2v[\cos(\varphi(t)+\pi), \sin(\varphi(t)+\pi)]^T$ and control point $\mathbf{P}_{3,j+1}$ satisfies the additional two conditions. Graph tree of the obtained paths is C2 continuous and more smooth as seen in Figs. 5 and 6.

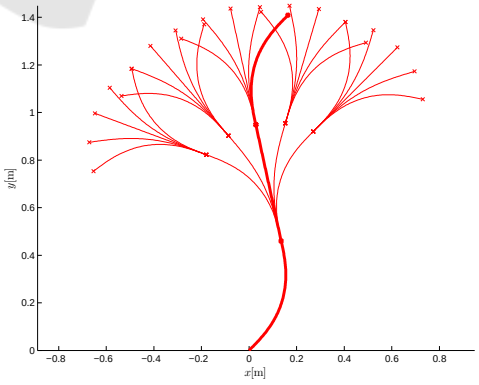

Figure 5: Search expansion using C2 continuous BB paths of fifth order.

The obtained combined path is therefore feasible for the robot to follow. It is smooth with continuous control velocities, continuous path curvature

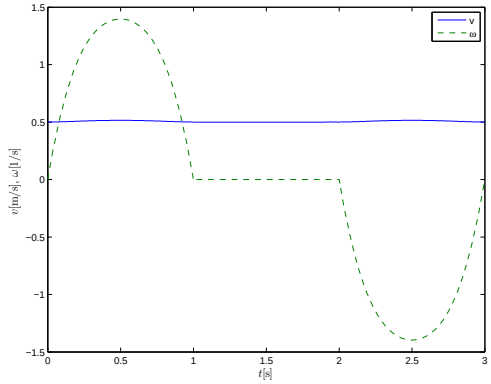

Figure 6: Differential drive robot control signal to follow thick path in Fig. 5.

and therefore is appropriate for optimal path-planing methods.

3 OPTIMAL PATH AND OPTIMAL VELOCITY PROFILE

The problem of finding shortest time optimal path in the environment with obstacles considering robot capabilities and obstacles is computationally intense. Therefore it usually is decoupled to a problem of finding a feasible collision safe path in discrete search space (e.g. A star algorithm) and then followed by some additional optimization.

Supposing the combined path from BB curves given in section 2 is the output of some path finding algorithm. The path is collision safe and close to being spatially optimal (depending on path discretization, e.g. number of defined successor curve segments). It additionally has continuous curvature which allow optimization of its velocity profile to achieve also shortest travelling time. Optimal velocity profile can be calculated as follows.

The curve is defined spatially by a schedule parameter u as $x_p(u)$, $y_p(u)$, $u \in [0,n]$ where n is the number of BB curves in the spline. The j-th curve of the spline is defined by scheduling parameter values $u \in (j-1, j]$ and maps to the j-th BB curve normalized time $\lambda_j = u - j + 1$ (valid if $j-1 \leq u \leq j$). To optimize the velocity profile of the given path one need to consider robot capabilities such as maximum velocity and acceleration which provide safe driving without slip of the wheels. To obtain the trajectory $x(t)$, $y(t)$ from the spatially given path one need to define the schedule $u = u(t)$. The curve translational velocities v, angular velocity ω and curvature κ are

$$v(t) = \sqrt{x_p'(u(t))^2 + y_p'(u(t))^2}\,\dot{u}(t) = v_p(u)\,\dot{u}(t) \tag{9}$$

$$\begin{aligned}\omega(t) &= \frac{x_p'(u(t))y_p''(u(t)) - y_p'(u(t))x_p''(u(t))}{x_p'(u(t))^2 + y_p'(u(t))^2}\dot{u}(t) \\ &= \omega_p(u)\,\dot{u}(t)\end{aligned} \tag{10}$$

$$\kappa(t) = \frac{x_p'(u(t))y_p''(u(t)) - y_p'(u(t))x_p''(u(t))}{\left(x_p'(u(t))^2 + y_p'(u(t))^2\right)^{3/2}} = \kappa_p(u) \tag{11}$$

where primes stand for derivatives with respect to u, while dots stand for derivatives with respect to t.

Main idea is to limit overall acceleration

$$a = \sqrt{a_t^2 + a_r^2} \tag{12}$$

which result is robot motion without wheel slipping (Lepetič et al., 2003). Where translational acceleration is $a_t = \frac{dv}{dt}$ and radial acceleration is $a_r = v\omega = v^2\kappa$. Maximal tangential a_{MAXt} and radial acceleration a_{MAXr} (can be different) define the overall acceleration to be somewhere on the ellipse

$$\frac{a_t^2}{a_{MAXt}^2} + \frac{a_r^2}{a_{MAXr}^2} = 1 \tag{13}$$

for time-optimal planning.

Robot translational velocity need to be the slowest in the curve points $u = u_{TPi}$ ($i = 1, \cdots, n_{TP}$, n_{TP} is number of TPs), called turn points (TP) where absolute value of the curvature is locally maximal. For all located TPs it holds: $a_t(u_{TPi}) = 0$ and $a_r u_{TPi} = a_{MAXr}$. Meaning that translational velocity in TP is defined as follows

$$v_p(u_{TPi}) = \sqrt{\frac{a_{MAXr}}{|\kappa(u_{TPi})|}} \tag{14}$$

and robot therefore need to decelerated before each TP and accelerate after each TP considering acceleration constraint (13). For each TP a candidate maximum velocity profile is computed and the final optimal velocity profile solution is obtained by minimizing all TPs candidate velocity profile. Because $v(t)$ and $v_p(u)$ are proportional dependant with time derivative of the schedule $\dot{u}(t)$ one can minimize the derivative of the schedule $\dot{u}(t)$ instead which is computed as follows. From acceleration constraint (13) considering

$$a_r(t) = v_p^2(u)\,\kappa_p(u)\,\dot{u}^2(t) \tag{15}$$

and

$$a_t(t) = \frac{dv_p(u)}{du}\dot{u}^2(t) + v_p(u)\,\ddot{u}(t) \tag{16}$$

follows the optimal schedule differential equation

$$\ddot{u} = \pm a_{MAXt} \sqrt{\frac{1}{v_p^2} - \frac{v_p^2 \kappa_p^2 \dot{u}^4}{a_{MAXr}^2}} - \frac{dv_p}{du} \frac{1}{v_p^2} \dot{u}^2 \qquad (17)$$

which solution is found numerically by integrating backward and forward in time from the TPs. Minus applies when deaccelerating (integrating backward in time) and positive sign when accelerating (integrating forward in time). Initial conditions $u(t)$ and $\dot{u}$ are defined by known position of TPs u_{TPi} and from

$$\dot{u}|_{TPi} = \sqrt{\frac{a_{MAXr}}{v_p^2(u_{TPi})\,\kappa_p(u_{TPi})}} \qquad (18)$$

knowing that radial acceleration in TP's is maximal allowable (only positive solution of (15) is used because $u(t)$ is strictly increasing function). The differential equations (17) are solved for each TP until the acceleration constrained is valid. The solution of the differential equation (17) therefore consists of the segments of $\dot{u}$ around each turning point

$$\dot{u}_l = \dot{u}_l(u)\,, \quad u \in [\underline{u}_l, \overline{u}_l]\,, \quad l = 1, \cdots, n_{TP} \qquad (19)$$

where $\dot{u}_l = \dot{u}_l(u(t))$ is the derivative of the schedule depending on u and $\underline{u}_l, \overline{u}_l$ are the l-th segment borders. Here the segments in (19) are given as a function of u although the simulation of (17) is done with respect to time. This is because time offset (time needed to arrive in TP) is not jet known, what is known at this point is relative time interval corresponding to each segment solution $\dot{u}_l$. Solution is possible if the union of all TP's solution intervals covers the whole interval of interest $[u_{SP}, u_{EP}] \subseteq \bigcup_{l=1}^{n_TP} [\underline{u}_l, \overline{u}_l]$ where start and end point are defined by $u_{SP} = 0$ and $u_{EP} = n$, respectively.

Final solution for $\dot{u}$ minimizes all partial solutions

$$\dot{u} = \min_{1 \le l \le n_{TP}} \dot{u}_l(u) \qquad (20)$$

and the time of the schedule $u(t)$ is obtained from $\dot{u}(u(t)) = \frac{du}{dt}$ as follows

$$t = \int_{u_{SP}}^{u_{EP}} \frac{du}{\dot{u}(u)} = t(u) \qquad (21)$$

Note that for time-optimal solution the travelling velocity as well as $\dot{u}(u)$ always are higher than 0. If $\dot{u}(u) = 0$ the system would stop which can not be time optimal solution.

To the velocity profile planning also requirements for initial v_{SP}, $v_p(u_{SP})$, and final velocities v_{EP}, $v_p(u_{EP})$ can be included. Starting point (SP) and end point (EP) can simply be treated as other turn points, their initial conditions reads $\dot{u}_{SP} = \frac{v_{SP}}{v_p(u_{SP})}$, $\dot{u}_{EP} = \frac{v_{EP}}{v_p(u_{EP})}$. Optimal solution only exist if these initial $\dot{u}_{SP}$ $\dot{u}_{EP}$ are both larger or equal than the solution for $\dot{u}$ obtained from TPs.

Additionally constraint on the maximum allowable velocity v_{MAX} ($v(t) <= v_{MAX}$) of the system should also be imposed. Whenever (during integrating (17)) the velocity constraint is violated the system need to stop accelerating and continue with velocity $v(t) = v_{MAX}$, meaning that schedule derivatives need to be set as follows $\ddot{u} = 0$ and $\dot{u} = \frac{v_{MAX}}{v_p(u)}$.

3.1 Examples of Optimal Velocity Profile Calculation

Taking path example from Fig. 5 where its velocity profile need to be optimized. Control points of the three BB curves are given as follows. The first curve is defined by

$$\mathbf{P}_{0,1} = [0,0]^T \quad \mathbf{P}_{1,1} = [0.0707, 0.0707]^T$$
$$\mathbf{P}_{2,1} = [0.1414, 0.1414]^T \quad \mathbf{P}_{3,1} = [0.1776, 0.2646]^T$$
$$\mathbf{P}_{4,1} = [0.1563, 0.3623]^T \quad \mathbf{P}_{5,1} = [0.1350, 0.4600]^T,$$

the second curve is defined by

$$\mathbf{P}_{0,2} = [0.1350, 0.4600]^T \quad \mathbf{P}_{1,2} = [0.1137, 0.5577]^T$$
$$\mathbf{P}_{2,2} = [0.0924, 0.6554]^T \quad \mathbf{P}_{3,2} = [0.0711, 0.7532]^T$$
$$\mathbf{P}_{4,2} = [0.0498, 0.8509]^T \quad \mathbf{P}_{5,2} = [0.0285, 0.9486]^T$$

and the third curve is defined by

$$\mathbf{P}_{0,3} = [0.0285, 0.9486]^T \quad \mathbf{P}_{1,3} = [0.0072, 1.0463]^T$$
$$\mathbf{P}_{2,3} = [-0.0141, 1.1440]^T \quad \mathbf{P}_{3,3} = [0.0221, 1.2672]^T$$
$$\mathbf{P}_{4,3} = [0.0928, 1.3379]^T \quad \mathbf{P}_{5,3} = [0.1635, 1.4086]^T$$

(all numbers are in meters). Additionally start and end velocity requirements for the combined path are $v_{SP} = 0.2\ m/s$, $v_{EP} = 0.1\ m/s$.

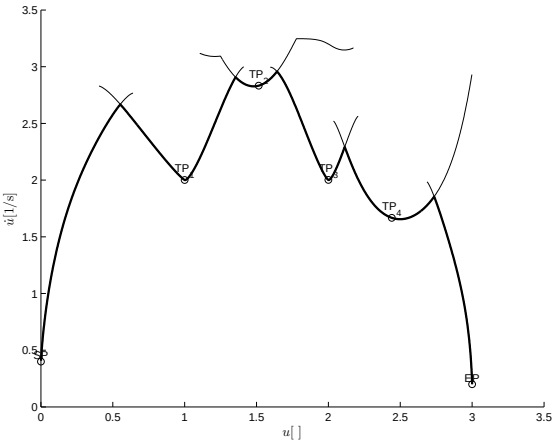

Figure 7: Optimal schedule minimizes all local profiles $\dot{u}$ in the turn points ($a_{MAXr} = 3$, $a_{MAXt} = 4$, $v_{MAX} = \infty$).

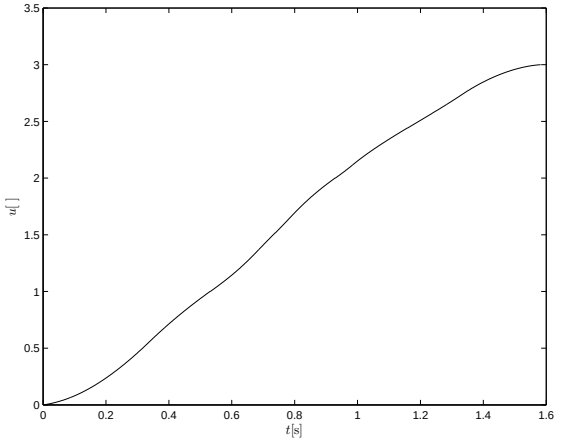

Figure 8: Optimal schedule $u(t)$ for the combined path ($a_{MAXr} = 3$, $a_{MAXt} = 4$, $v_{MAX} = \infty$).

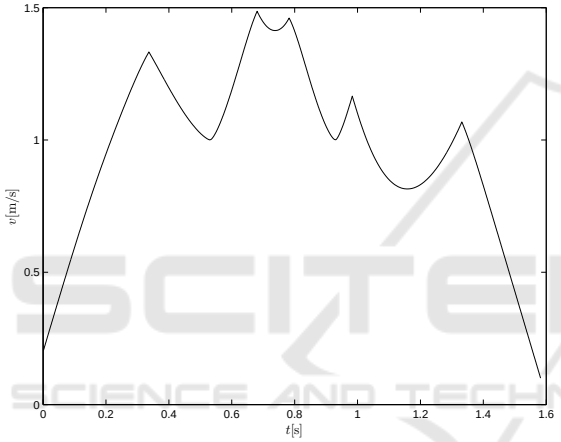

Figure 9: Optimal velocity profile $v(t)$ ($a_{MAXr} = 3$, $a_{MAXt} = 4$, $v_{MAX} = \infty$).

The optimal velocity profile for the combined path is first computed for acceleration constraints $a_{MAXr} = 3m/s^2$ and $a_{MAXt} = 4m/s^2$ and no velocity constraint.

Calculated optimal time derivative of the schedule parameter u along the path is shown by the thick line in Fig. 7. Where it is seen that minimum off all local turn points (TP) profiles and (SP) start and end point (EP) is selected. The resulting optimal schedule $u(t)$ is given in Fig. 8 and final velocity profile in Fig. 9

To simulate maximum driving velocity constraint $v_{MAX} = 1.3m/s$ the resulting optimal velocity profile changes as shown in Figs. 10- 11.

If translational acceleration is limited to $a_{MAXt} = 1.5m/s^2$ the optimal velocity profile considering acceleration and velocity constraints does not exist as the second turn point becomes unreachable as shown in Fig. 12.

Therefore new feasible maximal velocity (initially set by (18)) for the second TP need to be modified by

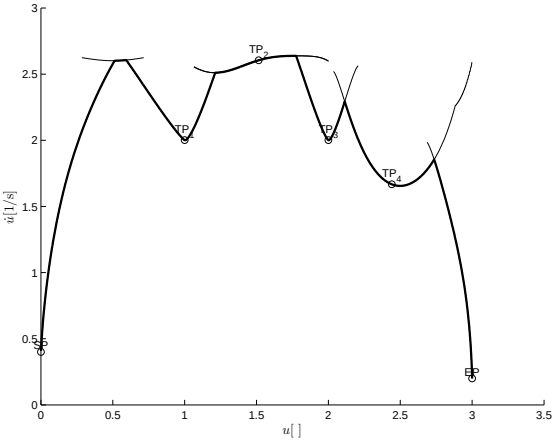

Figure 10: Optimal schedule $\dot{u}$ ($a_{MAXr} = 3m/s^2$, $a_{MAXt} = 4m/s^2$, $v_{MAX} = 1.3m/s$).

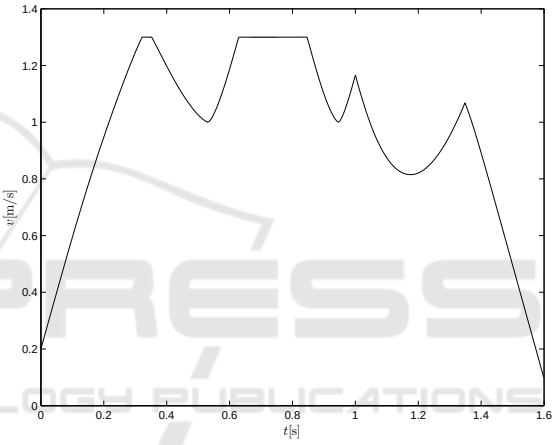

Figure 11: Optimal velocity profile $v(t)$ for $a_{MAXr} = 3m/s^2$, $a_{MAXt} = 4m/s^2$, $v_{MAX} = 1.3m/s$.

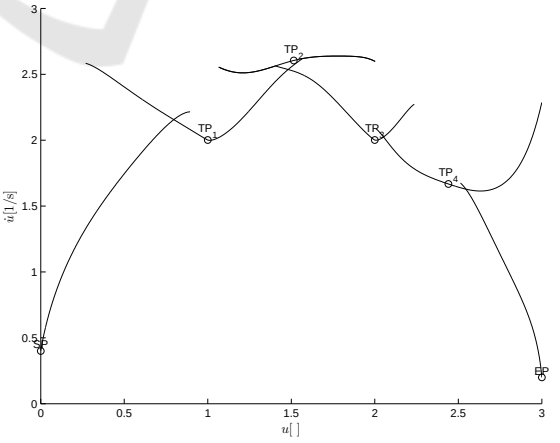

Figure 12: Unreachable optimal schedule. It is not possible to arrive in the second turn point with required schedule velocity ($\dot{u}_{TP_2} = 2.6\ 1/s$ at $u_{TP_2} = 1.5$) by acceleration $a_{MAXt} = 1.5m/s^2$.

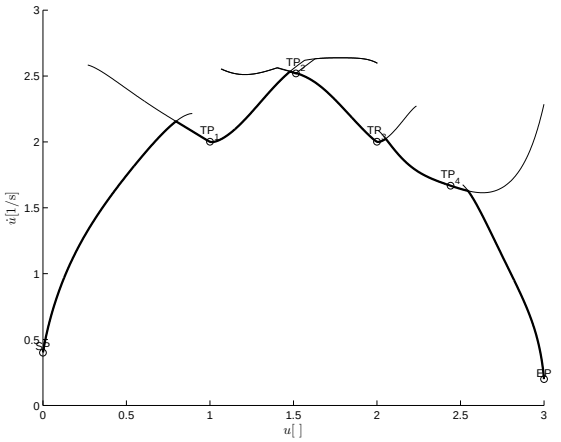

Figure 13: Optimal schedule $\dot{u}$ for $a_{MAXr} = 3m/s^2$, $a_{MAXt} = 1.5m/s^2$, $v_{MAX} = 1.3m/s$.

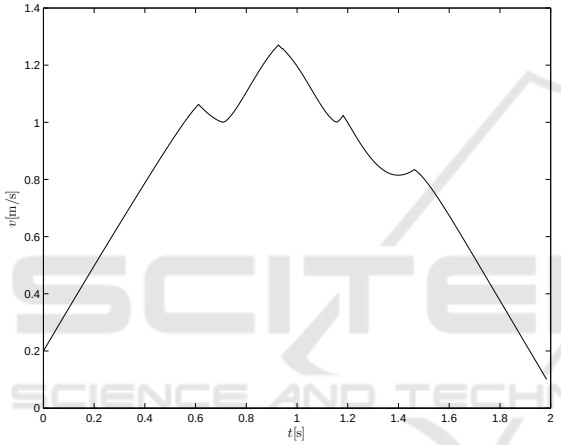

Figure 14: Optimal velocity profile $v(t)$ for $a_{MAXr} = 3m/s^2$, $a_{MAXt} = 1.5m/s^2$, $v_{MAX} = 1.3m/s$.

taking minimal off all local profiles at the $u_{TP_2} = 1.5$ which reads $\dot{u}_{TP_2} = 2.5\ 1/s$. The resulting optimal profile is then feasible as shown in Figs. 13-14.

4 CONCLUSION

The use of the fifth-order Bernstein-Bézier are proposed as the path sections comprising robot path in a hybrid path planing approaches. To obtain evenly spread of the path section candidates in each node end points of the sections and their orientations are pre computed assuming constant translational and angular velocity. From those final locations together with smooth transition requirements between the sections the Bernstein-Bézier polynomials are defined. For obtained composed path also an optimal velocity profile optimization approach is illustrated. The proposed approaches can be applied to a continuous path planing algorithm to find continuous curvature path with no additional smoothing required. Future issues will deal with computational complexity where velocity profile determination is integrated in the path planing. To obtain more optimal trajectories with shorter travelling time also variable final orientation of the path section candidates will be considered.

REFERENCES

Berglund, T., Erikson, U., Jonsson, H., Mrozek, K., and Soderkvist, I. (2011). Automatic generation of smooth paths bounded by polygonal chains. In *Proc. of the International Conference on Computational Intelligence for Modelling, Control and Automation (CIMCA)*, pages 528–535, Las Vegas.

Brezak, M. and Petrović, I. (2014). Real-time approximation of clothoids with bounded error for path planning applications. *IEEE Transactions on Robotics*, 30(2):507–515.

Chen, C., He, Y., Bu, C., Han, J., , and Zhang, X. (2014). Quartic Bezier Curve based Trajectory Generation for Autonomous Vehicles with Curvature and Velocity Constraints. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6108–6113, Hong Kong.

Choi, J.-W., Curry, R., and Elkaim, G. (2010). *Machine Learning and Systems Engineering*, chapter Piecewise Bezier Curves Path Planning with Continuous Curvature Constraint for Autonomous Driving, pages 31–45. Lecture Notes in Electrical Engineering 68. Springer Science+Business Media.

Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L., and Thrun, S. (2005). *Principles of robot motion: theory, algorithms, and implementations*. MIT Press.

Dolgov, D., Thrun, S., Montemerlo, M., and Diebel, J. (2008). Practical Search Techniques in Path Planning for Autonomous Driving. pages 1–6.

Dubins, L. (1957). On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. *Amer. J. Math.*, 79:497–516.

Kallmann, M. (2005). Path Planning in Triangulations. In *Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, International Joint Conference on Artificial Intelligence (IJCAI)*, pages 49–54, Edinburgh, Scotland.

Klančar, G., Zdešar, A., Blažič, S., and Škrjanc, I. (2017). *Wheeled mobile robotics - From fundamentals towards autonomous systems*. Elsevier, Butterworth-Heinemann.

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

LaValle, S. M. (2006). *Planning algorithms*. Cambridge University Press.

Lepetič, M., Klančar, G., Škrjanc, I., Matko, D., and Potočnik, B. (2003). Time optimal path planning

considering acceleration limits. *Robotics and Autonomous Systems*, 45:199–210.

Schwartz, J. and Sharir, M. (1990). *Algorithms and Complexity*, chapter Algorithmic motion planning in robotics, pages 391–430. Elsevier.

Sencer, B., Ishizaki, K., and Shamoto, E. (2015). A curvature optimal sharp corner smoothing algorithm for high-speed feed motion generation of NC systems along linear tool paths. *Int J Adv Manuf Technol*, 76(9):1977–1992.

Sethian, J. (1999). Fast Marching Methods. *SIAM Review*, 41(2):199–235.

Yang, X. and Wushan, C. (2015). AGV path planning based on smoothing A* algorithm. *International Journal of Software Engineering and Applications (IJSEA)*, 6(5):1–8.