Continuous-Curvature Bounded Trajectory Planning Using Parametric Splines

Mohamed ELBANHAWI [1,a], Milan SIMIC [a] and Reza JAZAR[a]
[a] *School of Aerospace, Mechanical and Manufacturing Engineering, RMIT University, Melbourne, Australia*

Abstract. This paper investigates trajectory generation for curvature continuous paths. A car-like wheeled robot platform is considered. Planning is performed using a single B-spline curve that maintains continuity through the path. Constraints such as bounded curvature, velocity, and acceleration limits are considered. We minimize the traversal time while maintaining continuity and obeying the robot's constraints. Simulation experiments were successfully conducted to verify the efficiency and robustness of this approach.

Keywords. Robot, Planning, Motion, Continuous Curvature, Bounded Curvature, Trajectory, Path.

Introduction

1.1. Path Planning

Autonomous robots are intelligent agents are capable of performing tasks independently. There is boundless potential for their applications. Planning is the process of finding a sequence of feasible actions to achieve a task. In the case of wheeled mobile robots the main task is reaching the goal location or a configuration. Path planning is a fundamental research area in robotics [1] and perhaps the most extensively studied [2]. Exact roadmaps, such a Visibility Graph and Voronoi Diagram, attempt to capture the connectivity of the search space. Cell Decompositions subdivide the search space and connect the free cells. These methods are computational extensive, especially in highly dimensional, or constrained scenarios. Potential field methods [3] guide the robot towards its goal by applying attractive forces towards the goal and repulsive, pushing away from the obstacles. Despite their effectiveness, potential field methods suffer from being trapped in local minima and oscillations [4]. Graph search algorithms, such as A* and AD* [5, 6] algorithms, are disadvantaged by discretizing the search space. Similarly, planning using state lattices [7, 8] relies on the discretization of the search space. In order to capture the entire search space high resolutions discretization is needed, leading to increased computations. On the other hand, low-resolution discretization leads to the loss of completeness, as the planner cannot guarantee the return of valid solutions in all environments. Sampling-based planning algorithms, which sample configuration space and attempt to capture its

[1] Corresponding Author: SAMME, Bundoora East Campus, RMIT University, Plenty Road, Bundoora, VIC 3083, Melbourne, Australia; Email: mohamed.elbenhawi@rmit.edu.au

connectivity, operate in continuous search spaces and thus they guarantee completeness. The most commonly used algorithms are rapidly-exploring random trees (RRT) [9] and probabilistic roadmap method (PRM) [10]. Sampling based planning is now an attractive area of robotic research especially kinodynamic, real-time and optimal planning in highly dimensional spaces [11].

The majority of planners produce straight-line path segments. Some classes of robots such as differential-drive, or omnidirectional robots, are capable of traversing those paths. However, car-like robots and fixed wing aerial vehicles, are incapable of following such paths, as they require stationary turning. Straight lines are an inefficient method of planning, as the robot needs to decelerate to stop, turn and then proceed to the next turn. Another disadvantage of these paths is their discontinuous trajectories leading to overactuation, slipping, mechanical wear and localization errors. Early attempts of smooth path generation, such as Dubin's and Reed and Shepp's [12] paths, relied on the amalgamation of lines and circular arcs. The resulting paths were discontinuous, difficult to compute and discretized the control space of the robot. Generating curvature continuous paths, that obey the vehicle's kinematics constraints, velocity and acceleration limits, is still a topic that is actively pursued. It is essential that the path planning method exhibit properties that enable real-time planning and re-planning in dynamic scenarios.

1.2. Related Work

B-spline curves enable the synthesis of curvature continuous paths. A genetic algorithm was used to select the shape of the path [13]. The number of control points had to be determined during the planner design, which limited its use. Another proposed planner was using B-spline for differential-drive robot and did not consider curvature constraints, or continuity [14]. A probabilistic B-spline was implemented using a greedy RRT algorithm [15]. Similarly, several smoothing algorithms were suggested for static environments [16, 17]. An efficient B-spline smoothing algorithm was recently proposed for sampling based planning [18]. It employs multiple B-spline segments, but it does not guarantee continuity at all instances.

Anderson, et al. [19] proposed the merger of circular arcs for UAV planning. This planner produced discontinuous paths and was limited to two-dimensional planar scenarios. Clothoids were proposed to overcome the discontinuity of circular arc paths [20]. Clothoids may seem like a suitable selection for robot path planning as they have continuous curvature. They are often used in road design due to the property of naturally increasing curvature. Unfortunately, Clothoids have no closed-form, and cannot be generated accurately in real-time, without limiting their length or orientation [21].

Bezier curves, like B-spline curves, are parametric curves that have also been implemented in robotic path planning. Quintic Beziers were used for kinodynamic planning with velocity and acceleration limits [22]. This approach did not consider the curvature continuity, or higher order constraints. The work of Kwangjin [23, 24] employed the planar geometric continuity conditions of Beziers [25] for robot path planning. It did not consider the initial and final headings of the vehicle. Three-dimensional points were mapped into a plane as the algorithm was limited to a single plane. It only guaranteed geometric continuity and did not provide any solution for velocity, or acceleration continuity, at the joining point between two Bezier segments.

Navigational task based comparative analysis was conducted to highlight the advantages of B-splines over other parametric curves [26]. B-spline can be locally modified without changing the entire path. Their order is independent of the control points. Consequently, a single curve segment can be used to represent an entire path. Using single segment approach guarantees curvature continuity at all sections. The properties of B-spline will be highlighted in section 3. Additionally we have introduced earlier methods to control B-spline curvature at sharp turns by relating the turning angle to the path curvature. Maneuvers have been introduced to handle the initial and final heading of the vehicle. Some of our previous findings will be shown in Section 3. Trajectory planning, while maintaining continuity under curvature, velocity, and acceleration limits, is addressed in section 4 and validated using simulation experiments in section 5.

2. B-Spline Curves

B-splines are parametric, vector-valued curves [27]. They are commonly used in Computer-Aided Design [28]. Recently, their application has emerged in robotics due to the effectiveness of their use in real-time planning [29] and complex maneuvers[30]. Numerical optimization is often employed to find the parameters of the curve that obey the path constraints [16, 17, 22, 31, 32]. Some of the drawbacks of these methods include that, curve parameters, such as the number of control points, must be predefined to ensure real-time performance. Otherwise, optimization is conducted offline while assuming a static environment [13, 15-17]. Another difficult task is controlling, or bounding the curvature of the trajectory.

A p-th degree B-spline curve, $c(u)$, defined by n control points and , $m = n+p+1$, knots $\hat{u}$, is given in equation (1), where u is the normalized path length parameter [28],

$$c(u) = \sum_{i=0}^{n} N_{n,i}(u) P_i \tag{1}$$

and where n is the number of control points. $N_{n,i}(u)$ is the B-spline basis function which is defined using the Cox-de Boor algorithm [33],

$$N_{i,p} = \begin{cases} 1 & u \in [\hat{u_i}, \widehat{u_{i+1}}) \\ 0 & else \end{cases} \tag{2}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{3}$$

The control points, P= $\begin{bmatrix} x_1, x_2 \dots x_n \\ y_1, y_2 \dots y_n \end{bmatrix}$, are elements of a set of waypoints that can be obtained by any planning algorithm. When subsequent control points are connected using straight lines they form a control polygon. Unlike Bezier curves, the order of a B-spline curve is independent on the number of control points, n, used to define it. Basis functions have local control on the curve, which allows for local modification of any segment of the path, without affecting the neighboring segments, or changing the shape of the entire curve, as shown in **Figure 3**(bottom).

3. Path Curvature

3.1. Continuous Curvature

A path characterized with a continuous curvature ensures smooth robot navigation. This smoothness reduces mechanical wear, prevents overshooting and facilitates feedback control in the execution stage [16, 23, 31]. The problem of continuity arises when connecting two different segments such as an arc and a line [12], two Beziers [23, 24] or two B-splines [18]. The advantage of B-spline over Beziers is that modifying waypoints have no effect on the degree of the curve, as already mentioned. It is possible to use a single segment to represent an entire path. This guarantees path continuity. We have proposed midpoint insertion between successive control points. This forces the curve's tangency to every side of the control polygon, as shown in **Figure 1**(a) and **Figure 3**(top).

3.2. Upper Bounded Curvature

Car-like vehicles and fixed wing UAVs are among a class of robots that have a limited turning radius. They are incapable of performing stationary turns so they must follow curved paths. The curvature of those paths is upper bound such that the robot does not exceed its turning radius. Curvature control of B-spline is a challenging task to be conducted using numerical optimization. An advantage of midpoint insertion is that the six-point segment, as shown in **Figure 1**(a) and **Figure 3**(top) is repeated throughout the path. L is the length of a segment side and α is the angle between both sides. It is possible to relate parameters of that segment to the corresponding B-spline curve. The control polygon can be formulated in terms of the segment parameters as given in equations (4) and (5).

$$P_x = \left[L, \quad \frac{L}{2}, \quad 0, \quad \frac{L}{2}\cos \propto, \quad L\cos \propto\right] \tag{4}$$

$$P_y = \left[0, \quad 0, \quad 0, \quad \frac{L}{2}\sin \propto, \quad L\sin \propto\right] \tag{5}$$

The parameter selection of the curve and the derivation of its equation are given in [26]. Cox-de Boor recursive algorithm in equations (2) and (3) is used to obtain the basis function using the curve parameters. Basis functions along with control points, equations (4) and (5), are substituted in equation (1) to obtain the curve equations, $c(u) = \begin{bmatrix} y(u) \\ x(u) \end{bmatrix}$. The curve is now defined in terms of the corresponding segment parameters. The derivatives of the curve, *x'(u), y'(u), x''(u), y''(u)* must be derived to compute the curvature of the B-spline segment, equation (6). Finally, the curvature can be expressed as a function of segment parameters, $k=f(L, \alpha)$, as given in equation (7).

$$k = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}} \tag{6}$$

$$k = \frac{2(u \sin \propto)(1-u)}{3L(2u^2(1-\cos \propto)(u^2-2u+1)+(2u-1)^2)^{3/2}} \quad (7)$$

It is possible to control the curvature of the path by modifying segment parameters. Adjusting the particular segment parameters will only have a local effect and will not modify the other path segments, which is an advantage of B-splines. Algorithms used to modify segment parameters, such that they satisfy the curvature constraints, headings and turning angles, have been presented in [26]. However, that analysis did not consider velocity and acceleration limits imposed on the vehicle.

4. Trajectory Planning

A path consisting of straight lines joining different waypoints, is smoothed using a clamped B-spline curve, *c(u)*. The curve is subdivided into different reoccurring segments whose parameters can be related to the curvature of the path. A single B-spline curve is used to ensure curvature continuity. It is required to plan the trajectory for the given curve, *c(u)*. The maximum limits for linear velocity, acceleration and angular velocity, v_{max}, a_{max}, ω_{max}, can be derived [34], and must not be exceeded. Additionally, initial and final velocity constraints must be obeyed. Minimizing the path traversal time, *T,* while maintaining the above conditions is desirable.

Consider a path consisting of a single clamped cubic B-spline curve, $c(u) = [x(u)\ y(u)]^T$, continuous curvature, *k(u)*, **Figure 1**. It will be shown, that a continuous and smooth velocity trajectory, *v(u)*, can be obtained. The path is subdivided into *N* equal *u* axis segments of size, Δu. The number of path segments, *N*, is carefully chosen to ensure the shape of the path is maintained. Using Fast Fourier Transform (FFT) the maximum path frequency component, f_{path}, is determined. Then N is selected such that the sampling frequency, $f_{sample} = \frac{1}{N}$, exceeds double the maximum path frequency. Assuming that the subdivision is close to be infinitesimal, *u*, is the normalized path length and *t* is the time parameter. It is possible to formulate all constraints, v_{max}, a_{max}, ω_{max}, v_o and v_f, in relation to an allowable continuous velocity, *v(t)*.

When the vehicle is turning, the linear velocity, *v*, is limited by the path curvature, *K*, and the maximum allowable angular velocity, ω_{max}, as shown in equation (8). At any instance, v_{max}, cannot be exceeded, as shown in equation (9). The initial and terminal velocities must be considered as well, as shown in equations (10) and (11) respectively. The initial and final conditions of the vehicle are often assumed to be static, i.e. $v_o = v_f = 0$. In some cases, such as re-planning or re-localization, the initial planning velocity may differ.

$$v(u) \leq \frac{\omega_{max}}{K} \quad (8)$$

$$v(u) \leq v_{max} \quad (9)$$

$$v(u) = v_o, \text{ when } u=0 \quad (10)$$

$$v(u) = v_f, \text{ when } u=1 \quad (11)$$

Considering the path, *c(u)* shown in **Figure 1**(a), that represents a robot traversing a single corner with continuous curvature, *k(u)* shown in **Figure 1**(b). The angular speed constraint, given by equation (8), is shown as a dotted line in **Figure 1**(c), while the linear velocities constraint, equation (9), is shown as a dashed line. It can be noticed that the robot must slow down as the curvature of the path increases. At u=0.42 in **Figure 1**(b), the curvature of the path reaches its maximum value, k_{max} which corresponds to the lowest allowable linear velocity through the corner, shown in **Figure 1**(c). Initial and final velocity conditions, equation (10) and (11), are shown in **Figure 1**(d). The robot starts with $v(0) = v_o$, and accelerates to reach $v(0.42) = \frac{\omega_{max}}{Kmax}$ and then decelerates towards a complete stop $v(1) = v_{f.}$ as shown in a solid red trajectory in **Figure 1**(d).

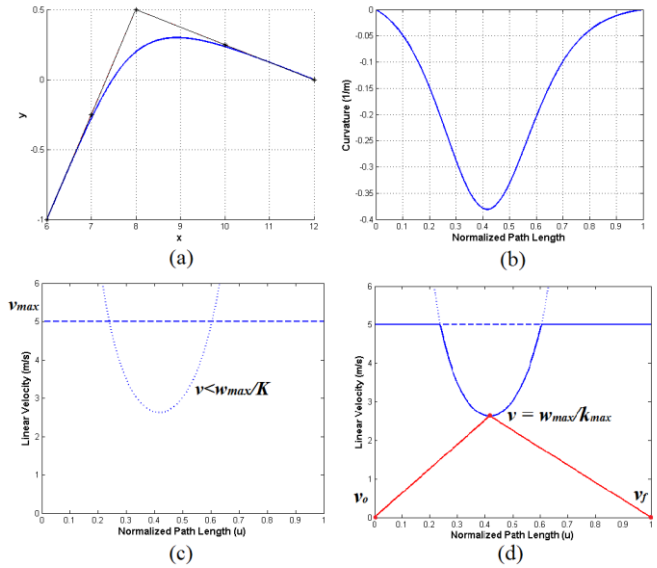

Figure 1. (a) A B-spline path *c(u)* (b) It's continuous curvature *k(u)* (c) The maximum allowable velocity due to linear (dashed line) and angular velocity (dotted) constraints (d) The resulting trajectory that accelerates from initial velocity towards the maximum velocity at *u*=0.42 and then decelerates towards to reach it the final desired velocity.

The obtained trajectory disregards acceleration constraint, a_{max}. Assuming infinitesimal subdivision, acceleration, *a(u)*, velocity, *v(u)*, displacement, *s(u)* and time interval Δt at any path segment of normalized length, Δu , can be defined as follows,

$$s(u) \approx \sqrt{\Delta x^2 + \Delta y^2} \tag{12}$$

$$v(u) = \frac{d(s(u))}{dt} \approx \frac{\Delta s(u)}{\Delta t} \tag{13}$$

$$\Delta t = \frac{\sqrt{\Delta x^2 + \Delta y^2}}{\Delta v(u)} \tag{14}$$

$$T = \sum_0^T \Delta t \tag{15}$$

$$a(u) = \frac{d(v(u))}{dt} \approx \frac{\Delta v(u)}{\Delta t} \tag{16}$$

The acceleration constraint, a_{max}, at any instance throughout the path can be defined as follows,

$$-a_{max} \leq \frac{\Delta v(u)}{\Delta t} \leq a_{max} \tag{17}$$

Iterating through every path segment, from $u=0$ to $u=1$, the acceleration condition in equation (17) is maintained as given by equation (18) and (19). This results in modifying the velocity graph, as shown in **Figure 2**. The initial part of the diagram, dotted line, exceeded the acceleration constraint and was modified. However, when decelerating the deceleration does not exceed $-a_{max}$ and is left unchanged.

$$\text{If } a(u) > a_{max}, \text{ set } v(u+\Delta u) = v(u) + a_{max}.\Delta t \tag{18}$$

$$\text{Else If } a(u) < -a_{max}, \text{ set } v(u+\Delta u) = v(u) - a_{max}.\Delta t \tag{19}$$

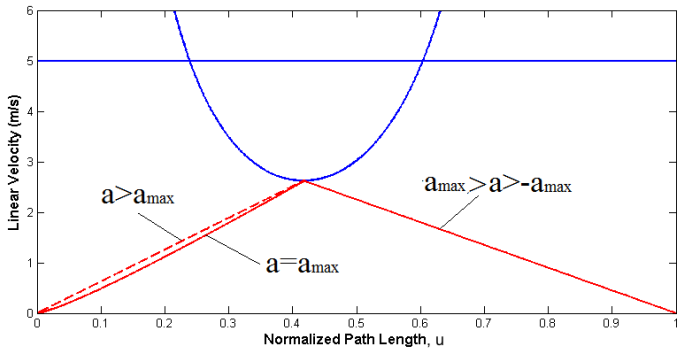

Figure 2. The original trajectory (dotted line) exceeds the allowable acceleration and is modified (solid line)

5. Simulation Experiments

Approach presented in the section (4) is illustrated and tested in two different scenarios. Let us assume that a path planning algorithm returns set of six subsequent waypoints, as given in **Table 2**. The algorithm proposed in [26] and highlighted in section (3) is used to generate a smooth cubic clamped B-spline curve. The B-spline path and its corresponding curvature are shown in **Figure 3** (top). The maximum allowable values and initial values are given **Table 1**. The resulting trajectories are given with respect to time, t, in **Figure 4**. The total traversal time is *33.35 seconds*.

The average runtime for both B-spline and trajectory planning is *0.047 seconds*[2]. Subsequently, this algorithm can run constantly, in the real time, while the robot is localizing itself and detecting obstacles for efficient re-planning purposes. To illustrate the robustness of this approach we assume that the robot localizes itself outside the desired path at x=3m and y=0m with v=2m/s. The algorithm is capable of re-planning

[2]on a 3.1 GHz i7 machine with 8GB of Memory running Windows 7 implemented in Matlab.

another B-spline path with continuous curvature as shown in **Figure 3** (bottom). The trajectory is regenerated from the current velocity of 2m/s to follow the original path and reach $v=0$ at the end of the path as shown in **Figure 4** (right).

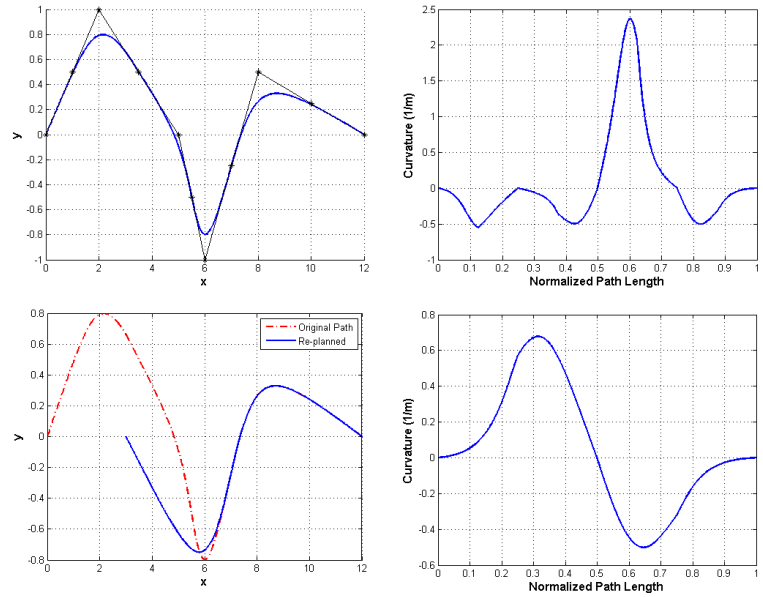

Figure 3. Initial cubic B-spline curve, c(u), and its corresponding continuous curvature, k(u), on top and the re-planned path and its curvature are shown at the bottom.

Table 1. Experiment Parameters

Parameters	Initial	Re-planning
v_{max} (m/s)	10	10
a_{max} (m/s^2)	5	5
ω_{max} (rad/s)	1	1
v_o (m/s)	0	**2**
v_f (m/s)	0	0

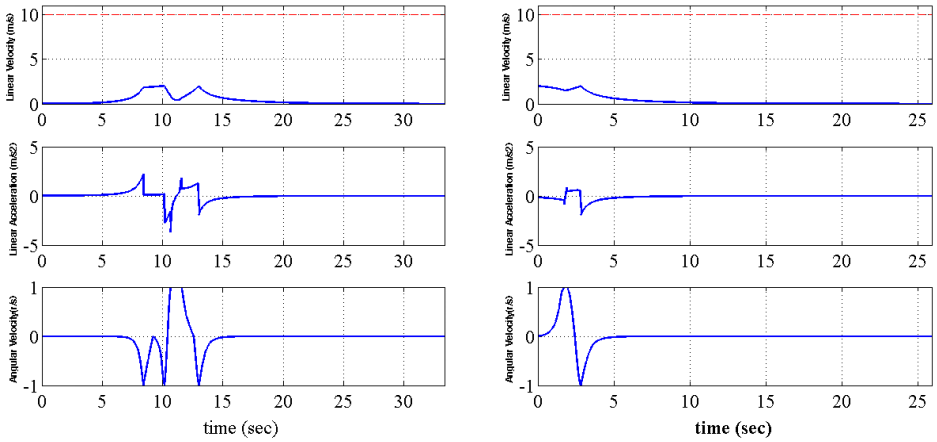

Figure 4. The linear velocity, v, angular velocity, ω, and linear acceleration, a, trajectories for the initial path (left) and re-planned path trajectories (right)

Table 2. Path Waypoints

	Initial						Re-planning			
Waypoint	**1**	**2**	**3**	**4**	**5**	**6**	**1**	**2**	**3**	**4**
x	0	2	5	6	8	12	3	6	8	12
y	0	1	0	-1	0.5	0	0	-1	0.5	0

6. Conclusion

This paper investigates trajectory planning for a wheeled robot. We considered vehicles that are incapable of performing stationary turns. B-spline curves were used to generate feasible paths. It was then required to plan the trajectory of the vehicle. Velocity and acceleration limits are both imposed on the vehicle path planning. We propose a method of that ensures that the curvature continuity is maintained and constraints are all obeyed. Our work provides a method that enables constrained trajectory planning in a real time fashion whilst maintaining continuity. Automated agricultural tractors, MAVs (micro aerial vehicles) with energy constraints due to battery sizes, passenger-carrying vehicles or sensitive cargo are some of the potential applications for our proposed approach. Some of the future works improving the implementation of the algorithm to achieve higher computational efficiency. Considering aerial vehicles trajectory planning and the different constraints imposed on such platforms. At this stage this approach is a performed post planning, including velocity, acceleration and curvature constraints will improve the performance of the planner by guiding the search directly towards more feasible regions.

References

[1] H. M. Choset, *Principles of Robot Motion: Theory, Algorithms, and Implementation*: Prentice Hall of India, 2005.

[2] J.-C. Latombe, "Motion Planning: A Journey of Robots, Molecules, Digital Actors, and Other Artifacts," *The International Journal of Robotics Research,* vol. 18, pp. 1119-1128, November 1, 1999.

[3] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *International Journal of Robotics Research,* vol. 5, pp. 90-98, Spr 1986.

[4] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, 1991, pp. 1398-1404 vol.2.

[5] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *Systems Science and Cybernetics, IEEE Transactions on,* vol. 4, pp. 100-107, 1968.

[6] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime search in dynamic graphs," *Artificial Intelligence,* vol. 172, pp. 1613-1643, 2008.

[7] M. Pivtoraiko, R. A. Knepper, and A. Kelly, "Differentially constrained mobile robot motion planning in state lattices," *Journal of Field Robotics,* vol. 26, pp. 308-333, 2009.

[8] M. Pivtoraiko and A. Kelly, "Kinodynamic motion planning with state lattice motion primitives," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011, pp. 2172-2179.

[9] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Iowa State University TR 98-11, 1998.

[10] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *Robotics and Automation, IEEE Transactions on,* vol. 12, pp. 566-580, 1996.

[11] M. Elbanhawi and M. Simic, "Sampling-Based Robot Motion Planning: A Review," *Access, IEEE,* vol. 2, pp. 56-77, 2014.
[12] J. A. Reeds and L. A. Shepp, "Optimal Paths For A Car That Goes Both Forward And Backward," *Pacific Journal of Mathematics,* vol. 145, pp. 367-393, Oct 1990.
[13] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on,* vol. 33, pp. 898-912, 2003.
[14] S. Gulati and B. Kuipers, "High performance control for graceful motion of an intelligent wheelchair," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 2008, pp. 3932-3938.
[15] E. Koyuncu and G. Inalhan, "A probabilistic B-spline motion planning algorithm for unmanned helicopters flying in dense 3D environments," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 815-821.
[16] T. Berglund, A. Brodnik, H. Jonsson, M. Staffanson, and I. Soderkvist, "Planning Smooth and Obstacle-Avoiding B-Spline Paths for Autonomous Mining Vehicles," *Automation Science and Engineering, IEEE Transactions on,* vol. 7, pp. 167-172, 2010.
[17] T. Maekawa, T. Noda, S. Tamura, T. Ozaki, and K.-i. Machida, "Curvature continuous path generation for autonomous vehicle using B-spline curves," *Computer-Aided Design,* vol. 42, pp. 350-359, 2010.
[18] J. Pan, L. Zhang, and D. Manocha, "Collision-free and smooth trajectory computation in cluttered environments," *The International Journal of Robotics Research,* vol. 31, pp. 1155-1175, September 1, 2012.
[19] E. P. Anderson, R. W. Beard, and T. W. McLain, "Real-time dynamic trajectory smoothing for unmanned air vehicles," *Control Systems Technology, IEEE Transactions on,* vol. 13, pp. 471-477, 2005.
[20] T. Fraichard and A. Scheuer, "From Reeds and Shepp's to continuous-curvature paths," *Robotics, IEEE Transactions on,* vol. 20, pp. 1025-1035, 2004.
[21] M. Brezak and I. Petrovic, "Real-time Approximation of Clothoids With Bounded Error for Path Planning Applications," *Robotics, IEEE Transactions on,* vol. PP, pp. 1-9, 2013.
[22] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 2427-2433.
[23] Y. Kwangjin and S. Sukkarieh, "An Analytical Continuous-Curvature Path-Smoothing Algorithm," *Robotics, IEEE Transactions on,* vol. 26, pp. 561-568, 2010.
[24] Y. Kwangjin, D. Jung, and S. Sukkarieh, "Continuous curvature path-smoothing algorithm using cubic Bezier spiral curves for non-holonomic robots," *Advanced Robotics,* vol. 27, pp. 247-258, 2013.
[25] D. J. Walton, D. S. Meek, and J. M. Ali, "Planar G2 transition curves composed of cubic Bézier spiral segments," *Journal of Computational and Applied Mathematics,* vol. 157, pp. 453-476, 2003.
[26] M. Elbanhawi and M. Simic, "Spline Framework for Autonomous Vehicles Navigation," *Robotica (under review),* 2013.
[27] I. J. Schoenberg, "Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions - A. On the Problem of Smoothing or Graduations a 1st Class of Approximation Formulae," *Quarterly of Applied Mathematics,* vol. 4, pp. 112-141, 946.
[28] G. Farin, *Curves and Surfaces for CAGD*: Morgan Kaufmann, 2002.
[29] E. Dyllong and A. Visioli, "Planning and real-time modifications of a trajectory using spline techniques," *Robotica,* vol. 21, pp. 475-482, 2003.
[30] M. Elbanhawi and M. Simic, "Examining the use of B-spline in Parking Assist Systems," *Applied Mechanics and Materials,* vol. 490-491, pp. 1025-1029, 2014.
[31] E. Magid, D. Keren, E. Rivlin, and I. Yavneh, "Spline-Based Robot Navigation," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, 2006, pp. 2296-2301.
[32] S. Gulati, C. Jhurani, B. Kuipers, and R. Longoria, "A framework for planning comfortable and customizable motion of an assistive mobile robot," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 4253-4260.
[33] C. De Boor, "On calculating with B-splines," *Journal of Approximation Theory,* vol. 6, pp. 50-62, 1972.
[34] K. G. Jolly, R. Sreerama Kumar, and R. Vijayakumar, "A Bezier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits," *Robotics and Autonomous Systems,* vol. 57, pp. 23-33, 2009.