Optimizacijski problem generiranja časovno minimalne trajektorije z uporabo Bézierovih krivulj

Martina Benko Loknar, Gregor Klančar, Sašo Blažič

Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška cesta 25, SI-1000 Ljubljana, Slovenija
E-mail: martina.loknar@fe.uni-lj.si

Povzetek

Članek obravnava problem časovno minimalnega planiranja gladkih trajektorij za avtonomne mobilne sisteme. Izračunana gladka pot je definirana po delih in jo sestavljajo Bézierove krivulje. Hitrostni profili na posameznih segmentih so časovno minimalni in imajo na spojih uporabljenih krivuljnih primitivov zvezno hitrost in pospešek. Rešitev iščemo z izračunavanjem časa potovanja vzdolž segmentov Bézierovih krivulj z nekaterimi prostimi parametri. Pri tem uporabimo algoritem, ki vzdolž dane poti generira časovno minimalen hitrostni profil z upoštevanjem omejitev hitrosti, pospeška in trzaja.

1 Uvod

Algoritmi planiranja poti imajo v mobilni robotiki izjemno široko uporabo in generirajo geometrijsko pot od začetne do končne točke prek vnaprej definiranih točk. Glavni cilj teh algoritmov je najti optimalno pot, ki (1) minimizira razdaljo med začetno in končno točko, (2) se izogne oviram in (3) je (dovolj) gladka. [1]. Algoritmi planiranja trajektorij pa točkam geometrijske poti pripišejo še časovne trenutke, ko bi naj jih mobilni sistem dosegel. Problem časovno minimalnega planiranja trajektorij ostaja aktualen zaradi naraščajočih zahtev po optimalnem delovanju mobilnih sistemov, robotov in avtomatiziranih strojev.

Definiranje časovnih trenutkov vzdolž poti vpliva na kinematične in dinamične lastnosti gibanja mobilnega sistema: sile in navori zavisijo od pospeška vzdolž trajektorije, medtem ko vibracije njegove mehanske strukture večinoma določajo vrednosti trzaja. Da bi zadostili kinematičnim omejitvam vozila, mora biti dobljena pot gladka; biti mora izvedljiva pri visokih hitrostih in hkrati neškodljiva za mehanski sistem v smislu izogibanja vibracijam in prevelikim pospeškom aktuatorjev [2]. Mnogo raziskav se zato posveča omejitvam ali minimizaciji trzaja na dani poti [3]. Planiranje trajektorije oziroma v splošnem planiranje gibanja mobilnih sistemov pa se pogosto razstavi na dva ločena problema: problem planiranja poti in problem planiranja hitrosti na tej dani poti [4]. Osnovni gradniki gladkih poti so krivuljni primitivi, kot so na primer polinomi, klotoide in Bézierove krivulje. Široka uporaba Bézierevih krivulj, ki jih definirajo t.i. kontrolne točke, je posledica njihovih ugodnih lastnosti [5]-[8]. Izbor kontrolnih točk pomembno vpliva na lastnosti generirane krivulje, zato so raziskave pogosto usmerjene v optimizacijo lege kontrolnih točk, tudi z uporabo metahevrističnih optimizacijskih algoritmov (npr., genetski algoritmi, optimizacija z roji delcev, ang. *tabu-search*, ang. *firefly algorithm*) [9][10]. Problem planiranja hitrosti na dani poti pa se ukvarja z generiranjem hitrostnega profila, ki je na dani poti izvedljiv. Kot omenjajo avtorji v [11], se v raziskavah pogosto predpostavi določeno obliko hitrostnega profila (polinom, trapez ipd.).

Članek obravnava problem generiranja časovno minimalnih trajektorij za kolesni mobilni sistem, ki se giba v omejenem prostoru brez ovir. Dirkalno stezo smo razdelili na več posameznih odsekov in na vsakem določili Bézierovo krivuljo, na kateri je bil čas vožnje najkrajši. Časovno minimalen hitrostni profil na posameznem krivuljnem primitivu smo določili z algoritmom, ki upošteva omejitve hitrosti, radialnega in tangentnega pospeška in radialnega in tangentnega trzaja.

2 Definicija problema

Naj bo gibanje delca vzdolž trikrat zvezno odvedljive planarne krivulje $\mathcal{C}$ opisano kot funkcija časa $t \in [0, t_f]$ s pozicijskim vektorjem $\mathbf{r}(t)$ z začetkom v danem fiksnem izhodišču. Vektorje hitrosti $\mathbf{v}(t)$, pospeška $\mathbf{a}(t)$ in trzaja $\mathbf{j}(t)$ lahko izrazimo v tangentno-normalni obliki kot:

$$\mathbf{v}(t) = v(t) \cdot \hat{\mathbf{T}} \tag{1a}$$

$$\mathbf{a}(t) = a_{\mathrm{T}}(t) \cdot \hat{\mathbf{T}} + a_{\mathrm{R}}(t) \cdot \hat{\mathbf{N}} \tag{1b}$$

$$\mathbf{j}(t) = j_{\mathrm{T}}(t) \cdot \hat{\mathbf{T}} + j_{\mathrm{R}}(t) \cdot \hat{\mathbf{N}}, \tag{1c}$$

pri čemer sta $\hat{\mathbf{T}}$ in $\hat{\mathbf{N}}$ enotski tangentni in enotski normalni vektor. Kolesni mobilni sistem ima nalogo, da v čim krajšem času prevozi dirkalno stezo v ravninskem prostoru brez ovir. Pri tem je njegovo gibanje dodatno omejeno z dinamičnimi omejitvami hitrosti, pospeška in trzaja:

$$0 \leq \|\mathbf{v}(t)\| \leq v_{\mathrm{MAX}}; \quad \forall t \in [0, t_f], \tag{2a}$$

$$\frac{a_T^2(t)}{a_{T_{\mathrm{MAX}}}^2} + \frac{a_R^2(t)}{a_{R_{\mathrm{MAX}}}^2} \leq 1; \quad \forall t \in [0, t_f], \tag{2b}$$

$$\frac{j_T^2(t)}{j_{T_{\mathrm{MAX}}}^2} + \frac{j_R^2(t)}{j_{R_{\mathrm{MAX}}}^2} \leq 1; \quad \forall t \in [0, t_f]. \tag{2c}$$

Naš cilj je razviti in implementirati algoritem za generiranje trajektorije, ki zadosti vsem naštetim omejitvam.

Algoritem, ki vzdolž dane poti generira časovno minimalen hitrostni profil z omejitvami hitrosti, pospeška in trzaja, je sestavljen iz dveh korakov. V prvem koraku algoritma (podrobneje opisan v [12]) se upoštevajo omejitve hitrosti in pospeška. V drugem koraku algoritem modificira prvotni hitrostni profil tako, da so upoštevane tudi omejitve trzaja, pri čemer se postopek razlikuje glede na tip kršitve (točkaste ali intervalne kršitve trzaja). Na danem izvedljivem segmentu poti, ki je v našem primeru Bernstein-Bézierova krivulja, je optimizacijski problem najti hitrostni profil $v(t)$, ki doseže konec krivulje v minimalnem času tako, da pri tem ni presežena nobena od omejitev. Eden od dosedanjih poskusov rešitve tega problema je opisan tudi v [13].

3 Krivuljni primitivi

N-dimenzionalen Bernsteinov polinom n-tega reda, $\mathbf{r}_n(\lambda) : [0,1] \to \mathbb{R}^N$, je definiran kot:

$$\mathbf{r}_n(\lambda) = \sum_{i=0}^{n} \mathbf{P}_{i,n} B_{i,n}(\lambda), \qquad \lambda \in [0,1] \tag{3}$$

pri čemer je λ normaliziran čas ($0 \leq \lambda \leq 1$), $\mathbf{P}_{i,n} \in \mathbb{R}^N$ je i-ta kontrolna točka in $B_{i,n}(\lambda)$ je baza Bernsteinovih polinomov, definirana kot:

$$B_{i,n}(\lambda) = \binom{n}{i} \lambda^i (1-\lambda)^{n-i}, \tag{4}$$

za vse $i \in \{0, \ldots, n\}$, pri čemer je $\binom{n}{i}$ binomski koeficient. Naj bo $\mathbf{P}_n = [\mathbf{P}_{0,n}, \ldots, \mathbf{P}_{n,n}] \in \mathbb{R}^{N \times (n+1)}$ vektor kontrolnih točk polinoma $\mathbf{r}_n(\lambda)$. Bernsteinov polinom v enačbi (3) lahko tedaj zapišemo v matrični obliki:

$$\mathbf{r}_n(\lambda) = \mathbf{P}_n \begin{bmatrix} B_{0,n}(\lambda) \\ B_{1,n}(\lambda) \\ \cdots \\ B_{n,n}(\lambda) \end{bmatrix} \tag{5}$$

Kadar se Bernsteinovi polinomi uporabljajo za opis planarnih in prostorskih krivulj, se jih pogosto imenuje tudi Bézierove krivulje. Zaradi ugodnih geometrijskih lastnosti in numerične stabilnosti se Bézierove krivulje pogosto uporabljajo pri načrtovanju poti mobilnih sistemov [5]. Prva in zadnja točka Bernsteinovega polinoma iz enačbe (3) sta tudi njuni končni točki:

$$\mathbf{r}_n(0) = \mathbf{P}_{0,n} \quad \text{and} \quad \mathbf{r}_n(1) = \mathbf{P}_{n,n}. \tag{6}$$

Poleg tega N-dimenzionalen Bernsteinov polinom n-tega reda leži notraj konveksne ogrinjače, ki jo definira $\mathbf{P}_n$. Začetek in konec Bézierove krivulje je tudi tangenten na prvi in zadnji odsek konveksnega poligona:

$$\left.\frac{\mathrm{d}\mathbf{r}_n}{\mathrm{d}\lambda}\right|_{\lambda=0} = n(\mathbf{P}_{1,n} - \mathbf{P}_{0,n}) \tag{7}$$

$$\left.\frac{\mathrm{d}\mathbf{r}_n}{\mathrm{d}\lambda}\right|_{\lambda=1} = n(\mathbf{P}_{n,n} - \mathbf{P}_{n-1,n}) \tag{8}$$

Preostale lastnosti Bernsteinovih polinomov (odvodi, računanje določenih integralov, de Casteljaujev algoritem, višanje stopnje polinomov ipd.) ne spadajo v vsebinskega okvir tega članka; za zainteresiranega bralca je več detajlov na voljo v [5].

3.1 Konstrukcija posamezne krivulje

Bézierove krivulje, ki jih določa zelo veliko število kontrolnih točk, so numerično nestabilne, zato je pri planiranju poti zaželjeno konstruirati gladko pot, ki jo sestavlja več Bézierovih krivulj nizkega reda. Na spojih posameznih krivulj mora biti izpolnjen pogoj zvezne ukrivljenosti. Najmanjši red Bézierovih krivulj, ki zadosti tej zahtevi je $n = 5$. Bézierova krivulja petega reda $\mathbf{r}_5(\lambda) = [x(\lambda), y(\lambda)]^{\mathrm{T}}$ je definirana s šestimi kontrolnimi točkami $\mathbf{P}_{i,5} = [x_i, y_i]$, $i \in \{0, 1, \ldots 5\}$:

$$\begin{aligned} \mathbf{r}_5(\lambda) = {} & (1-\lambda)^5 \mathbf{P}_{0,5} + 5\lambda(1-\lambda)^4 \mathbf{P}_{1,5} \\ & + 10\lambda^2(1-\lambda)^3 \mathbf{P}_{2,5} + 10\lambda^3(1-\lambda)^2 \mathbf{P}_{3,5} \\ & + 5\lambda^4(1-\lambda)\mathbf{P}_{4,5} + \lambda^5 \mathbf{P}_{5,5} \end{aligned} \tag{9}$$

Članek obravnava zgolj Bézierove krivulje petega reda, zato bomo v nadaljevanju članka zaradi preglednosti pri sklicevanju na kontrolne točke izpuščali drugi indeks, ki označuje red krivulj. Za učinkovito iskanje optimalne trajektorije je zelo pomemben način konstrukcije Bézierovih krivulj in izbor ustreznih optimizacijskih parametrov. Odvod ukrivljenosti določa vrednost radialnega trzaja (prvi člen v spodnjem izrazu):

$$\mathbf{j}(t) = \left(\ddot{v} - \frac{v^3}{\kappa^2}\right) \cdot \hat{\mathbf{T}} + \frac{1}{v}\left(\frac{\mathrm{d}}{\mathrm{d}t}\frac{v^3}{\kappa}\right) \cdot \hat{\mathbf{N}} \tag{10}$$

ki jo v $\mathbf{P}_0$ postavimo na 0 kot smiselno dodatno omejitev trzaja na spojih zlepkov. Koordinate posameznih kontrolnih točk $\mathbf{P}_i$ označimo z (x_i, y_i) za $i \in \{0, \ldots, 5\}$. Za $i \in \{0, \ldots, 4\}$ pa označimo razdalje med zaporednimi kontrolnimi točkami $d(\mathbf{P}_i, \mathbf{P}_{i+1})$ z d_{i+1} in kote $\sphericalangle(d_i, y = 0)$ s φ_i (slika 1). Velja:

$$x_{i+1} - x_i = d_{i+1} \cos\varphi_{i+1} \tag{11a}$$

$$y_{i+1} - y_i = d_{i+1} \sin\varphi_{i+1} \tag{11b}$$

Izraza 11a-11b ovrednotimo za $i \in \{0, 1\}$, uporabimo adicijske izreke in dobimo naslednji izraz za ukrivljenost κ_0 v $\mathbf{P}_0$:

$$\lim_{\lambda \to 0} \kappa(\lambda) = \kappa_0 = \frac{4}{5}\frac{d_2}{d_1^2} \sin(\varphi_2 - \varphi_1) \tag{12}$$

in za odvod ukrivljenosti κ_0' v $\mathbf{P}_0$:

$$\begin{aligned} \lim_{\lambda \to 0} \frac{\mathrm{d}\kappa}{\mathrm{d}\lambda} = {} & \frac{12}{5}\frac{1}{d_1^2} d_3 \sin(\varphi_3 - \varphi_1) + \\ & + \kappa_0 \left(-12\frac{d_2}{d_1}\cos(\varphi_2 - \varphi_1) + 6\right) \end{aligned} \tag{13}$$

Če drugi člen v izrazu (13) postavimo na 0, potem:

$$\frac{d_2}{d_1} = \frac{1}{2\cos(\varphi_2 - \varphi_1)} \tag{14}$$

Ukrivljenost in njen odvod v $\mathbf{P}_0$ sta tedaj:

$$\kappa_0 = \frac{4}{10}\frac{\tan(\varphi_2 - \varphi_1)}{d_1} \tag{15a}$$

$$\kappa_0' = \frac{12}{5}\frac{d_3 \sin(\varphi_3 - \varphi_1)}{d_1^2} \tag{15b}$$

Konstrukcija posamezne Bézierove krivulje tako poteka v naslednjih korakih (slika 1):

1. Začrtamo prvo kontrolno točko in jo označimo z $\mathbf{P}_0$. V smeri φ_1 odmerimo dolžino d_1 in označimo drugo točko z $\mathbf{P}_1$.

2. V smeri φ_1 nato odmerimo dolžino $d_2^{||}$, ki sledi iz izraza (14):

$$d_2^{||} = d_2 \cos(\varphi_2 - \varphi_1) = \tfrac{1}{2} d_1 \qquad (16)$$

3. V pravokotni smeri nato odmerimo razdaljo $d_2^{\perp}$, (izraz 12):

$$d_2^{\perp} = \tfrac{5}{4} d_1^2 \kappa_0 \qquad (17)$$

in tretjo kontrolno točko označimo s $\mathbf{P}_2$.

4. Vse točke, ki so od $\mathbf{P}_2$ v isti smeri (pravokotno na daljico $\overline{\mathbf{P}_0\mathbf{P}_1}$) oddaljene za $d_3^{\perp}$ (izraz 13), označimo s črtkano črto:

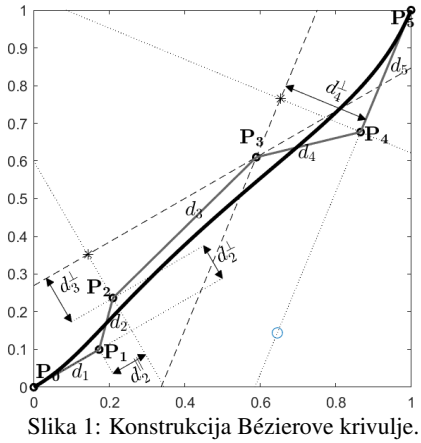
$$d_3^{\perp} = \tfrac{5}{12} d_1^2 \kappa_0' \qquad (18)$$

5. Začrtamo zadnjo kontrolno točko in jo označimo s $\mathbf{P}_5$. V smeri kota φ_5 odmerimo dolžino d_5 in peto kontrolno točko označimo s $\mathbf{P}_4$.

6. Vse točke, ki so od $\mathbf{P}_4$ oddaljene za $d_4^{\perp}$ (izrazi 11a-11b, 12 za $i = 4$) pravokotno glede na daljico $\overline{\mathbf{P}_4\mathbf{P}_5}$, označimo s črtkano črto:

$$d_4^{\perp} = \tfrac{5}{4} d_5^2 \kappa_5 \qquad (19)$$

7. Četrta kontrolna točka $\mathbf{P}_3$ leži na presečišču obeh črtkanih črt. Bézierova krivulja je sedaj popolnoma definirana.

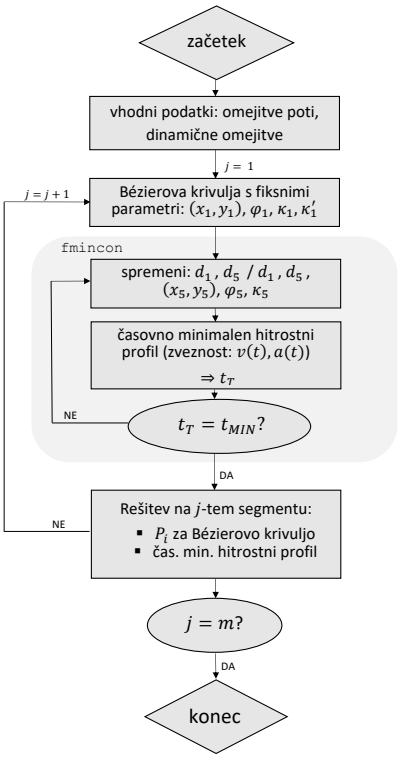


Slika 1: Konstrukcija Bézierove krivulje.

3.2 Generiranje celotne trajektorije

Kolesni mobilni sistem se po ravni podlagi lahko giba po dirkalni stezi – omejenem prostoru brez ovir. Da bi povečali učinkovitost optimizacije, dirkalno stezo razdelimo na posamezne odseke, na katerih iščemo optimalne trajektorije. Problem je zastavljen tako, da je na posameznem odseku steze vnaprej znan položaj (x_1, y_1) prve kontrolne točke Bézierove krivulje, kot φ_1, ter vrednosti ukrivljenosti κ_0 in odvoda ukrivljenosti κ_0' v $\mathbf{P}_0$.

Shema na sliki (2) opisuje splošno delovanje predstavljenega algoritma za generiranje trajektorij. Dirkalna steza je sestavljena iz m segmentov in j je trenutno zaporedno število segmenta, na katerem se izvaja računanje: $j \in \{1, 2, \ldots m\}$. Uporabljena nelinearna optimizacijska metoda je funkcija `fmincon` programskega okolja MATLAB, gradientna metoda, ki v vsaki iteraciji spremeni proste parametre krivulje tako, da se vrednost kriterijske funkcije (čas potovanja vzdolž krivulje) postopno zmanjšuje in naposled doseže minimum. Čas potovanja t_T vzdolž krivulje se izračuna z uporabo algoritma, ki generira časovno minimalen hitrostni profil. Pri tem je potrebno paziti, da so hitrosti in pospeški na spojih zlepkov zvezni. Postopek se ponavlja na vsakem segmentu, dokler ni določena celotna trajektorija vzdolž dirkalne steze.



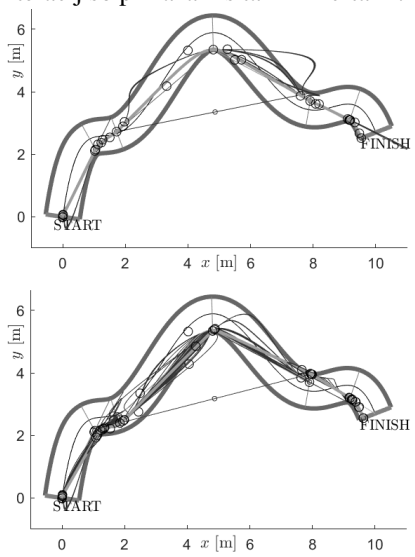
Slika 2: Konstrukcija Bézierove krivulje.

Algoritem za generiranje časovno minimalne trajektorije $\mathcal{T}$ je bil izveden v dveh različicah. V prvi različici z rešitvijo $\mathcal{T}_2$ sta prosta parametra le dva, razdalji d_1 in d_5, začetni približek optimizacije pa je čas potovanja vzdolž hevristično določene Bézierove krivulje. V drugi različici z rešitvijo $\mathcal{T}_{5,\mathrm{II}}$), pa je prostih parametrov pet: razdalji d_1, d_5, (x_5, y_5), kot φ_5 in κ_5. Začetni približek optimizacije ($\mathcal{T}_{5,\mathrm{I}}$) pa je rešitev optimizacijskega problema na istem odseku z dvema prostima parametroma.

4 Rezultati simulacij in komentarji

Simulacije smo izvedli v programskem okolju MATLAB. Na dani dirkalni stezi, razdeljeni na 6 odsekov, smo z gradientno metodo izračunali časovno minimalno trajektorijo za kolesni mobilni sistem z naslednjimi dinamičnimi omejitvami: $v_{\mathrm{MAX}} = 1.5\,\mathrm{m/s}$, $a_{\mathrm{T_{MAX}}} = 2\,\mathrm{m/s^2}$, $a_{\mathrm{R_{MAX}}} = 4\,\mathrm{m/s^2}$, $j_{\mathrm{T_{MAX}}} = 6\,\mathrm{m/s^3}$ and $j_{\mathrm{R_{MAX}}} = 8\,\mathrm{m/s^3}$.

Slika (3) prikazuje dano 6-segmentno dirkalno stezo in časovno minimalne trajektorije (debele črte). Vmesni rezultati iteracij so prikazani s tankimi črtami.



Slika 3: Iskanje časovno minimalnih trajektorij $\mathcal{T}_2$ (zgoraj) in $\mathcal{T}_{5,II}$(spodaj).

Tabela (1) prikazuje čase potovanja $t_i\,[s]$ na posameznih odsekih dirkalne steze. Trenutni delni rezultati naše študije so pokazali, da naj bi bil čas potovanja najkrajši v primeru optimizacije le dveh prostih parametrov, ki določata Bézierove krivulje. Poleg tega, da nastavitev možnosti optimizacije v funkciji `fmincon` zahteva veliko izkušenj, pa nekonsistentnost rezultatov lahko kaže na nezveznost kriterijske funkcije, kar je veliko bolj kompleksen, a ne nerešljiv problem.

	t_1	t_2	t_3	t_4	t_5	t_6	$\sum_{i=1}^{6} t_i$
$\mathcal{T}_2$	1,97	0,62	2,73	2,34	0,93	0,47	9,06
$\mathcal{T}_{5,\mathrm{I}}$	1,97	0,59	2,81	2,34	0,98	0,54	9,23
$\mathcal{T}_{5,\mathrm{II}}$	1,91	0,57	2,78	2,29	0,98	0,54	9,07

Tabela 1: t_i in celoten čas potovanja $\sum_{i=1}^{6} t_i$ za $\mathcal{T}$.

5 Zaključek

Članek predstavi delovanje algoritma za planiranje časovno minimalnih trajektorij v okolju brez ovir. Temelji na uporabi Bézierovih krivulj in upošteva prostorske (gibanje po dirkalni stezi v ravnini) in dinamične omejitve (hitrosti, pospeška in trzaja). Kljub nekaterim omejitvam predstavljene metode verjamemo, da bodo rezultati te študije lahko osnova za naše raziskovanje v prihodnosti.

V prihodnje se bomo najprej usmerili v razvoj in validacijo algoritma za generiranje časovno minimalnih trajektorij s hkratnim upoštevanjem poteka dirkalne steze na naslednjih odsekih (analogija s strategijo pomičnega horizonta, v ang. *receding horizon strategy*) in natančnejšo analizo učinkovitosti algoritma (računskih časov). Morda bi se kot ugoden način reševanja izkazala tudi uporaba Bézerovih krivulj višjega reda.

Zahvala

Članek je nastal s finančno podporo Agencije Republike Slovenije za raziskovalno dejavnost (št. P2-0219).

Literatura

[1] A. Tharwat, M. Elhoseny, A. E. Hassanien, T. Gabel, and A. Kumar, "Intelligent Bézier curve-based path planning model using Chaotic Particle Swarm Optimization algorithm," *Cluster Computing*, vol. 22, pp. 4745–4766, 2019.

[2] A. Gasparetta, P. Boscario, A. Lanzutti, and R. Vidoni, *Motion and Operation Planning of Robotic Systems*, vol. 29 of *Mechanisms and Machine Science*. Cham: Springer International Publishing, 2015.

[3] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A Review of Motion Planning Techniques for Automated Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[4] X. Qian, F. Altché, P. Bender, C. Stiller, and A. De La Fortelle, "Optimal trajectory planning for autonomous driving integrating logical constraints: An MIQP perspective," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 205–210, 2016.

[5] C. Kielas-Jensen and V. Cichella, "BeBOT: Bernstein Polynomial Toolkit for Trajectory Generation," *IEEE International Conference on Intelligent Robots and Systems*, no. 3, pp. 3288–3293, 2019.

[6] L. Zheng, P. Zeng, W. Yang, Y. Li, and Z. Zhan, "Bézier curve-based trajectory planning for autonomous vehicles with collision avoidance," *IET Intelligent Transport Systems*, vol. 14, no. 13, pp. 1882–1891, 2020.

[7] R. Cimurs, J. Hwang, and I. H. Suh, "Bezier curve-based smoothing for path planner with curvature constraint," *Proceedings - 2017 1st IEEE International Conference on Robotic Computing, IRC 2017*, pp. 241–248, 2017.

[8] H. Li, Y. Luo, and J. Wu, "Collision-free path planning for intelligent vehicles based on bézier curve," *IEEE Access*, vol. 7, pp. 123334–123340, 2019.

[9] G. Kamaras, P. Stamatopoulos, and S. Konstantopoulos, "Path planning for terrain of steep incline using Bezier curves," *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, vol. 2020-Novem, pp. 101–105, 2020.

[10] B. Li, L. Liu, Q. Zhang, D. Lv, Y. Zhang, J. Zhang, and X. Shi, "Path planning based on firefly algorithm and Bezier curve," *2014 IEEE International Conference on Information and Automation, ICIA 2014*, no. July, pp. 630–633, 2014.

[11] X. Qian, I. Navarro, A. de La Fortelle, and F. Moutarde, "Motion planning for urban autonomous driving using Bézier curves and MPC," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, vol. 2030, pp. 826–833, IEEE, nov 2016.

[12] G. Klančar, A. Zdešar, S. Blažič, and I. Škrjanc, *Wheeled Mobile Robotics: From Fundamentals Towards Autonomous Systems*. 2017.

[13] M. Loknar, G. Klančar, and S. Blažič, "Trajectory Planning By Applying Optimal Velocity Profile Algorithm on Bernstein-Bézier Motion Primitives," *29. Mednarodna Elektrotehniška in računalniška konferenca*, pp. 144–147, 2020.