Načrtovanje poti za avtonomno parkiranje vozila AGV

Kristina Nikolovska
Mentor: izr. prof. dr. Gregor Klančar
Univerza v Ljubljani, Fakulteta za elektrotehniko
Tržaška c. 25, 1000 Ljubljana, Slovenija
kristina.nikolovska6@gmail.com, gregor.klancar@fe.uni-lj.si

A path planning algorithm for parking an autonomous vehicle

Recently, with the development of self-driving, autonomous vehicles also called automated guided vehicles (AGV) and automated warehouses there have been a growing interest in developing autonomous path planning as well as parking algorithms. In this study, an algorithm presents a solution to a 4-wheel- autonomous vehicle parking. An arc path parking algorithm is proposed. The path is circle arc that connects the car initial position and goal position of the car inside the parking space. The main focus is given to the algorithm testing and presentation of obtained test results on simulation environment. At the end the good and the bad sides of the used algorithm are presented as well as some aspects of real implementation.

Kratek pregled prispevka

Z razvojem avtonomnih vozil oziroma avtomatiziranih vodenih vozil (automated guided vehicles, AGV) in avtomatiziranih skladiščih se je razvila tudi potreba po razvoju algoritmov za avtonomno planiranje poti kot tudi za parkiranje. V tej študiji bo prikazano reševanje problema parkiranja štirikolesnega vozila AGV. Eden od ključnih problemov pri procesu parkiranja je planiranje poti. V članku je predstavljen algoritem za iskanje poti s pomočjo krožnih lokov, ki povezuje začetno pozicijo vozička s ciljno pozicijo vozička. Glavni poudarek je na testiranju delovanja tega algoritma in predstavitvi rezultatov simulacije. Na koncu so predstavljene dobre in slabe lastnosti algoritma in nekateri vidiki, katere je potrebno upoštevati pri realni implementaciji.

1 Uvod

Z razvojem avtonomnosti mobilnih robotov [1][2], se je pojavila tudi potreba za razvoj boljših algoritmov avtonomnega vodenja. Eno od osnovnih opravil pri mobilni robotiki je planiranje poti, ki rešuje problem iskanja optimalne oz. najboljše poti mimo ovir iz začetne do ciljne točke. Za planiranje poti v znanem ali neznanem prostoru, obstajajo številne splošne in računsko učinkovite metode iskanja optimalne poti. Med njimi sta bolj znani A*, Dijkstra [3]. Za bolj specifične probleme, kot je primer avtonomnega parkiranja, pa potrebujemo bolj natančne algoritme, ki planirajo pot tudi do milimetra natančno in pri planiranju učinkovito upoštevajo dimenzije vozička, parkirnega prostora in lahko tudi dimenzije tovora na vozičku [3],[4],[5]. Konkretni primer, kjer je avtonomno parkiranje potrebno je, ko s pomočjo vozila AGV v tovarnah premikamo in vozimo voziček za prenos tovora.

Planiranje poti predstavlja en del celotne sheme izvedbe avtonomnega parkiranja mobilnega robota v določen parkirni prostor, poleg lokalizacije in vodenja. V tem prispevku bo predstavljen algoritem za planiranje poti s pomočjo krožnih lokov [3], ki se lahko vključi v celoten proces avtonomnega parkiranja mobilnega robota. Predstavljena bo implementacija, analiza delovanja, optimizacijska študija izbire nastavitvenih parametrov in testiranje tega algoritma. Delovanje samega algoritma je prikazano na simulacijskem modelu, ki je narejen v okolju Matlab. Podani so rezultati simulacije in njihovo vrednotenje.

2 Opis problema

Problem, na katerega se osredotoča ta raziskava, je parkiranje vozila AGV, ki deluje v tovarnah in pomaga pri prevažanju tovornih vozičkov. AGV vozilo se večinoma zapelje pod tovorni voziček, ga prime in odpelje na željeno mesto. Gibanje sklopa AGV-ja in tovornega vozička v predpisano parkirno mesto imenujemo parkiranje. Samo parkiranje je sestavljeno iz več opravil, ki sestavljajo celotni proces.

Obstajata lahko dve različni nalogi. Prva, ko AGV dobi nalogo, da določen tovorni voziček odpelje na predpisano mesto, druga naloga pa, da se sam AGV parkira na določeno polnilno postajo.

Pri prvi nalogi se planiranje poti razdeli na več delov: prvo je planiranje poti iz trenutne pozicije AGV-ja v neposredno bližino trenutne pozicije vozička z materialom, oz. točke, kjer se nato izvaja algoritem za prevzem vozička. Drugi del je planiranje poti, kjer se AGV zapelje pod voziček in ga prime, tako da se lahko skupaj premikata naprej. Tretji del je planiranje poti, kjer se AGV skupaj z vozičkom premakne v bližino željene ciljne pozicije oz. točke, kjer se lahko začne postopek parkiranja. Četrti del planiranja je, da se AGV skupaj z vozičkom parkira na ciljno pozicijo. Peti in tudi zadnji del je, da AGV spusti voziček in se premakne, tako da je ponovno prost za naslednjo nalogo, oz. do točke, ki je definirana kot zadnja točka za izpetje AGV-ja. Pri tem se prvi in treti del planiranja lahko rešujeta z že dobro znanimi algoritmi za planiranje, kot je npr. A*. Medtem, ko ostali deli planiranja poti rabijo bolj natančno planiranje, ki upošteva dimenzije samega AGV in tovornega vozička in tudi dimenzije parkirnih prostorov.

Planiranje poti druge naloge pa je sestavljena samo iz dveh opravil in sicer: premik AGV iz trenutne pozicije v bližino parkirnega prostora oz. polnile postaje do točke, kjer se lahko začne parkirni proces in proces parkiranja na ciljno pozicijo. Za prvi del planiranja poti se zopet lahko izkoristijo že dobro znani algoritmi, kot je A*. Za drugi del naloge pa je potrebo bolj natančno planiranje, ki upošteva dimenzije samega AGV in parkirnega prostora.

Problem, na katerega se osredotoča ta raziskava je četrti del planiranje poti prve naloge, torej da se AGV skupaj s tovornim vozičkom parkira v parkirni prostor in drugi del druge naloge, da se sam AGV parkira v polnilno postajo. Vsi rezultati, ki si prikazani v poglavju Rezultati, so prikazani za reševanje

problema, ko se AGV zapelje v parkirni prostor (polnilna postaja). Predstavljen algoritem se lahko uporabi tudi za reševanje problema, ko se AGV skupaj s tovornim vozičkom zapelje v parkirni prostor.

3 Algoritem za planiranje poti s krožnimi loki

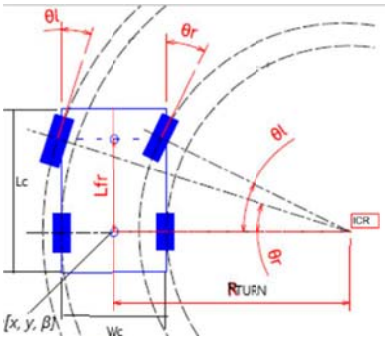
Parkirna pot je krožni lok, ki povezuje začetno pozicijo vozička s ciljno pozicijo. Da lahko parkira, mora voziček slediti tej poti po krožnem loku.

3.1 Model vozička

Model vozička (slika 1), ki je uporabljen za simulacijo in predstavitev delovanje tega algoritma je štirikolesnik s premikajočimi sprednjimi kolesi (Ackermannov pogon). V celotni študiji je lega vozička definirana z vektorjem P=[x, y, β], ki predstavlja točko v centru zadnje osi in orientacijo te osi β. Iz kinematičnega modela vozička lahko izračunamo θ (glej (1)), ki je povprečni krmilni kot vozička, če imamo podana zasuka levega in desnega prednjega kolesa (θ_l, θ_r). Izračunamo lahko tudi radij obračanja vozička *Rturn*, pri trenutnem kotu krmiljenja θ in razdalji L_{fr} med prednjo in zadnjo osjo, kot podaja enačba (2) in slika 1.

$$\theta = (\theta_l + \theta_r)/2 \qquad (1)$$

$$Rturn = L_{fr}/tan(\theta) \qquad (2)$$



Slika 1: Model vozička.

3.2 Izračun ICR centrov, Rmin in Rmax ter lokalni koordinatni sistem

ICR (ang. Instantaneous Center of Rotation) točka predstavlja trenuten center rotacije in leži na presečišču osi vseh koles. Definira torej točko, okoli katere krožijo vsa kolesa z enako krožno hitrostjo ω glede na *ICR (slika 1)*. Glede na zasuk koles v levo ali desno, se točka *ICR* izračuna po enačbi (3). Vsak voziček ima minimalni radij *Rmin*, ki je po notranji (bližji ICR) strani vozička, maksimalni radij *Rmax*, ki je na zunanji strani vozička in radij obračanja vozička *Rturn*, ki je radij glede na center vozička, v našem primeru je to center na zadnji osi. Vsi radiji in točka ICR se izračunajo po enačbi (3).

$$ICR = \begin{cases} [x - Rturn\sin\beta,\ y + Rturn\cos\beta] & če\ \theta < 0 \\ [x + Rturn\ \sin\beta, y - Rturn\cos\beta] & če\ \theta >= 0 \end{cases}$$

$$Rmin = Rturn - Wc\ /2 \qquad (3)$$

$$Rmax = \begin{cases} \sqrt{(Rturn + Wc/2)^2 + (Lc - Lfr)^2} & za\ vzratno \\ \sqrt{(Rturn + Wc/2)^2 + Lfr^2} & za\ naprej \end{cases}$$

Lokalni koordinatni sistem se definira s ciljem, da se poenostavijo izrazi. Vsi preračuni v algoritmu se dogajajo glede na ta lokalni koordinatni sistem. Lokalne koordinate (X0,Y0) so definirane z enačbo (4). V enačbah je *Wp* širina parkirnega mesta, *Wc* je širina vozička, *Lc* je dolžina vozička, *Lfr* je razdalja od sprednjega odbijača do zadnje osi, *Rturn* je najmanjši krog obračanja vozička, *Rmin*/*Rmax* pa so minimalni in maksimalni krog obračanje telesa vozička, glede na *Rturn*.

$$X0 = Wp/2$$

$$Y0 = \begin{cases} \sqrt{Rmax^2 - (Rmin + (\mathrm{Wp}/2)^2)^2} & če\ pogoj\ A \\ -\sqrt{(WcWp/2) - (Wc/4)^2} & če\ pogoj\ B \end{cases} \qquad (4)$$

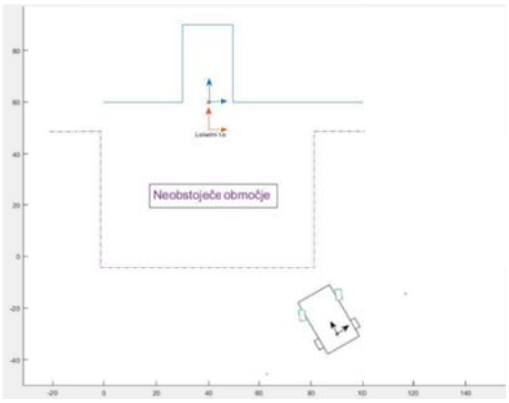
V enačbi (4) je pogoj *A* izpolnjen, ko je ($Rmax^2$-($Rmin$+$(Wp/2)^2)^2$)>=0 in pogoj *B* je izpolnjen, ko velja ($Rmax^2$-($Rmin$+$(Wp/2)^2)^2$)<0.

3.3 Postopek parkiranja

Algoritem uporablja krožne loke za premik od začetne pozicije vozička do ciljne pozicije. Pot, po krožnem loku ima to prednost, da omogoča majhne napake pri sledenju poti in se enostavno izračuna.

3.3.1 Nedovoljeno območje

Nedovoljeno območje je območje, v katerem poti za parkiranje ni mogoče najti. To se lahko zgodi zato, ker je voziček preblizu parkirnemu prostoru in ne more izvest premikov, ki so potrebni za parkiranje. Nedovoljeno območje se lahko določi na več načinov. Eden od teh načinov je eksperimentalno določanje, kjer se za različne pozicije vozička v bližini parkirnega prostora pogleda ali tam obstaja končni krog in se na ta način določijo meje, kjer se proces parkiranje ne mora začeti. Lahko se pa že na začetku določijo območja, ki niso primerna za parkiranje zaradi ovir, ali pa se določi, da se parkiranje začne na neki oddaljenosti od parkirnega prostora (npr. 8 m od parkirnega prostora). Nedovoljeno območje za ta primer je prikazano na sliki 2.



Slika 2: Prikaz nedovoljenega območja.

3.3.2 Izbira končne poti

Izbira smeri obračanja in smeri vožnje je odvisna od napovedi končne poti. Iz določene začetne pozicije vozička obstajajo štiri končne poti (rdeči krogi na sliki 3). Določijo so glede na zavijanje levo / desno in vožnjo naprej/nazaj. Končna pot je izbrana, če izpolnjuje dva pogoja. Prvi pogoj, ki mora bit izpolnjen, da je končna pot izbrana, je ta, da mora biti radij končnega kroga večji od minimalnega radija obračanja vozička. V tem primeru je radij krogov C3 in C4 večji kot *Rturn* vozička. Torej kroga C1 in C2 ne moremo izbrati kot končna. Drugi pogoj pa je, da je tangenta točka dotikališča med končnimi krožnicami (rdeče) in krožnicami z minimalnim radijem obračanja (zeleni) nad X osjo lokalnega koordinatnega sistema. Ta pogoj izpolnjujeta kroga C1 in C4. Torej krog C4 izpolnjuje oba pogoja, zato je ta krog izbran kot končni krog.

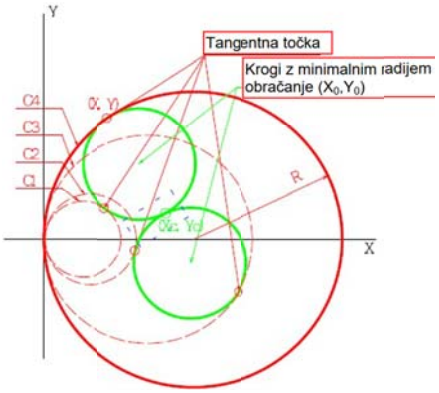
Enačbe, po katerih se izračunajo radij končnega kroga in tangenta točka dotikališča med končnimi krožnicami (rdeče) in krožnicami z minimalnim radijem obračanja (zeleno) *(X,Y)* so zapisane v (*5*). Pri tem je *X0*, *Y0* središče minimalnih krožnic obračanja, Rturn je radij obračanja vozička, *R* radij končnega kroga in *X* je *x* komponenta tangentne točke, *Y* je *y* komponenta tangentne točke.

$$B=X0^2 + Y0^2$$

$$R = (B + Rturn^2)/(2 * (Rturn + X0)) \qquad (5)$$

$$X = ((Rturn + X0) * (B - Rturn^2))/((Rturn + X0)^2 + Y0^2)$$

$$Y = (Y0 * (B - Rturn^2)/((Rturn + X0)^2 + Y0^2)$$



Slika 3: Prikaz vseh možnih končnih poti.

3.3.3 Osnovna izvedba algoritma

Osnovna izvedba algoritma predpostavlja znane pozicije, dimenzije in koordinatni sistem parkirnega prostora, znane so tudi dimenzije in lega AGV-ja. AGV je že v bližini parkirnega prostora in začetna pozicija AGV-ja je točka, v kateri se začne izvajati parkirni algoritem. Samo računanje v algoritmu se začne z določanjem θ

in *Rturn* AGV-ja. Naslednji korak je določitev lokalnega koordinatnega sistem. Preveri se ali je AGV znotraj nedovoljenega območja. Če AGV ni v tem območju, se izračuna točka *ICR* in radij *R* končnega kroga ter tangente točke (X,Y). Preveri se ali so pogoji za izbor končne poti izpolnjeni in če so, se sestavi referenčna pot, ki je rešitev algoritma. Če pa se AGV nahaja v nedovoljenem območju, potem pa se izračuna točka *ICR* in se namesto končne poti izbere enega od krogov z radijem obračanja *Rturn* za referenčno pot, po kateri se AGV premika za neko dolžino krožnega loka. Ko se premakne za to dolžino po krožnem loku pa se za novo pozicijo AGV-ja preveri ali je še v nedovoljenem območju.

3.3.4 Optimizacija pozicije in orientacije vozička

Ta optimizacija algoritma se izvede z namenom, da se preveri delovanje algoritma v različnih območjih v bližini parkirnega prostora in da se določi območje, ki ni primerno za začetek parkiranja in območje, ki je najbolj primerno. Pri tem se upošteva, da se parkiranje začne v znanem okolju z določenimi dimenzijami parkirnega prostora in vozička. Z optimizacijo orientacije vozička pa se analizira, kako vpliva orientacije na referenčno pot.

3.3.5 Optimizacija kota θ

Koti koles θ_l in θ_d in posledično povprečni krmilni kot vozička θ predstavljajo pomembne podatke za računanje poti. Z njegovim spreminjanjem se spreminja tudi radij obračanja vozila *Rturn*. S spreminjanjem tega radija se spreminja tudi tangentna točka dotikališča med končnim krogom in krogom z minimalnem radijem obračanja, kar rezultira v spreminjane dolžine na poti. Pri tej optimizaciji imamo podano referenčno pozicijo vozička za začetek parkiranja in iščemo, s kakšnim povprečnim kotom obračanja vozička θ lahko dosežemo najbolj optimalno pot.

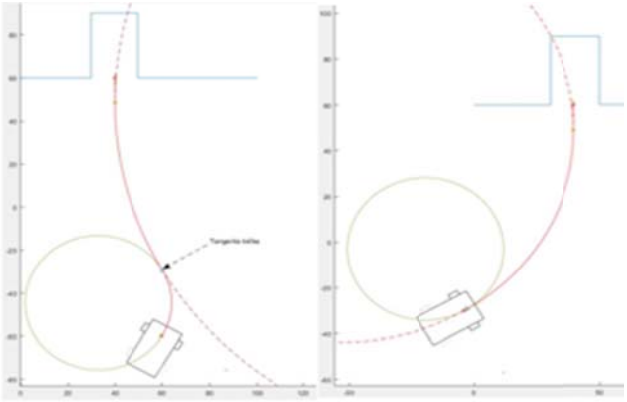
4 Rezultati

Testiranje algoritma je bilo izvedeno za različne pozicije vozička, različne orientacije vozička, za različne dimenzije vozička ter različne pozicije in dimenzije parkirnega prostora. Pri testiranju algoritma je bilo pokazano, da je algoritem pravilno izveden in da deluje. Algoritem vedno najde referenčno pot, če voziček ni preblizu parkirnemu prostoru oziroma znotraj nedovoljenega območja. Primer najdene poti je prikazan na sliki 4. Dolžina poti je odvisna tudi od parametra, ki predpiše parkiranja naravnost ali parkiranja vzvratno. Parkiranje naravnost da bolj optimalne (krajše) poti, če je voziček obrnjen s sprednjim delom proti parkirnemu prostoru. Vzvratno parkiranje pa rezultira v krajše poti, če je voziček z zadnjim delom obrnjen proti parkirnemu prostoru.

4.1.1 Optimizacija pozicije in orientacije vozička

Splošne ugotovitve, do katerih se je s pomočjo teh optimizacije prišlo so naslednje. Zato, da je pot optimalna (najkrajša), mora bit tangenta točka dotikališča med krogom obračanja vozička in končnim krogom v točki pozicije vozička. V tem primeru je pot najkrajša in je sestavljena iz premika po končnem krožnem loku in premikom naravnost. Tak primer je prikazan na sliki 4. Lahko se zgodi, da je tista pozicija in orientacija vozička, ki je določena za začetek parkiranja v nedovoljenem območju oz. preblizu parkirnemu prostoru, kot je prikazano na sliki 5 na levi strani. V tem primer se vseeno najde končna pot z dodatnim manevrom. Tak primer se reši tako, da se voziček premika po krogih obračanja stran od te pozicije. Lahko se določi korak premika, torej dolžino krožnega loka. Ko se voziček premakne za to dolžino, se ponovno izvede algoritem in se preračuna ali v novi točki najdena končna pot. Ta korak se lahko večkrat ponovi, dokler ni najdena taka točka, da je referenčna pot do parkirišča možna. Na sliki 5 je prikazan postopek, kjer se voziček premakne v točko *P_new* kjer se s pomočjo algoritma ponovno preračuna končna pot, ki je veljavna. V splošnem lahko za vsak primer vozička zaključimo, da se s spreminjanjem orientacije ne

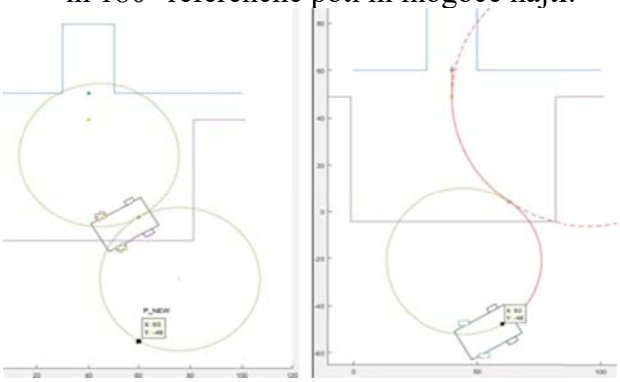
spreminja nedovoljeno območje (kjer referenčna pot ne obstaja), se pa spreminja dolžina poti.



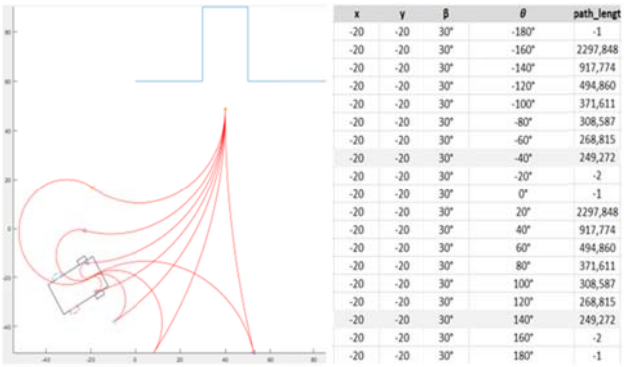
Slika 1: Na levi sliki prikaz končne poti in na desni sliki prikaz optimalne končne poti.

4.1.2 Optimizacija kota θ

S to optimizacijo se določi, kateri kot θ je najbolj optimalen. Na sliki 6 lahko vidimo kako se spreminja dolžina poti glede na izbiro kota θ. Z izvedbo te optimizacije za več različnih pozicij je bilo ugotovljeno, da pri kotih 0 °,-180 ° in 180° referenčne poti ni mogoče najti.



Slika 5: Prikaz primera, ko se AGV znajde v nedovoljenem območju, in ponovni izračun poti.



x	y	β	θ	path_lengt
-20	-20	30°	-180°	-1
-20	-20	30°	-160°	2297,848
-20	-20	30°	-140°	917,774
-20	-20	30°	-120°	494,860
-20	-20	30°	-100°	371,611
-20	-20	30°	-80°	308,587
-20	-20	30°	-60°	268,815
-20	-20	30°	-40°	249,272
-20	-20	30°	-20°	-2
-20	-20	30°	0°	-1
-20	-20	30°	20°	2297,848
-20	-20	30°	40°	917,774
-20	-20	30°	60°	494,860
-20	-20	30°	80°	371,611
-20	-20	30°	100°	308,587
-20	-20	30°	120°	268,815
-20	-20	30°	140°	249,272
-20	-20	30°	160°	-2
-20	-20	30°	180°	-1

Slika 6: Spreminjanje dolžino poti, glede na izbiro kota θ.

5 Zaključek

V tem članku smo se osredotočili na algoritem iskanje poti s krožnimi loki za namen parkiranja vozička. Algoritem prikazuje praktičen primer planiranje poti pri procesu parkiranja. Pokazano je bilo, da algoritem vedno najde pot in da je računalniško učinkovit.

Prednosti predlaganega algoritma so: upoštevanje dimenzij parkirnega prostora in dimenzij vozička, prilagojen je za različne dimenzije in modele vozička, lahko se prilagodi za različne dimenzije parkirnega prostora, če se po nekem naključju voziček znajde preblizu parkirnemu prostoru in če referenčne poti ni mogoče najti, se lahko z dodatnim manevrom to reši. Slabost pa je, da imamo v osnovni izvedbi algoritma samo eno končno pot, ki izpolnjuje omenjene pogoje. Prvi pogoj, ki mora biti izpolnjen, da je končna pot izbrana, je ta, da mora biti radij končnega kroga večji od minimalnega radija obračanja vozička. Drugi pogoj pa zahteva, da je tangenta točka dotikališča med končnim krogom in krogom z minimalnim radijem obračanja nad X osjo lokalnega koordinatnega sistema. Z izbiro take poti izgubimo fleksibilnost. Izbrana končna pot že sama določa ali bo vozilo parkiralo naravnost ali vzvratno in na katero stran bo prvi obrat (levo/desno). To je lahko slabost, če je v določenem primeru način parkiranja boljši kot drugi. Tudi na število manevrov ne moramo vplivati. Zato je lahko referenčna pot, ki jo dobimo ne-optimalna, glede na zahteve za določeni primer parkiranja.

Iskanje izvedljive poti je odvisno od vseh parametrov, torej pozicije, orientacije in kota koles vozička. Zato moramo imeti za lažjo izvedbo optimizacije poti, točno določen vsaj eden od teh parametrov. Za izračun končne poti imamo lahko nekaj scenarijev:

- pozicija in orientacija vozička za začetek parkiranja je že določena in iščemo optimalno pot iz te pozicije in orientacije,
- omejen ali določen je kot zasuka koles,
- določeno je območje, v katerem se začne parkiranje,

- določeno je parkiranje vzvratno ali naprej v parkirni prostor .

Delovanje algoritma je bilo analizirano s pomočjo različnih optimizacij, njihove ugotovitve so opisane v rezultatih prispevka.

6 Literatura

[1] G. Klančar, A. Zdešar, S. Blažič, I. Škrjanc, *Wheeled mobile robotics : from fundamentals towards autonomous systems,* Oxford; Cambridge (MA): Butterworth-Heinemann, cop. 2017.

[2] R. Siegwart and I. R. Nourbakhsh, *Introduction to autonomous mobile robots.* Cambridge, Mass: MIT Press, 2004.

[3] S. M. LaValle, *Planning Algorithms.* Cambridge: Cambridge University Press, 2006.

[4] Y. Wang, D. K. Jha, and Y. Akemi, *A two-stage RRT path planner for automated parking,* in 2017 13th IEEE Conference on Automation Science and Engineering (CASE), Xi'an, 2017, pp. 496–502.

[5] A. Giese, *A Comprehensive, Step-by-Step Tutorial on Computing Dubin's Curves, p. 21*.