Vodenje in lokalizacija mobilnega sistema v okolju

Marko Hančič, Aleš Šink, Andrej Zdešar, Gregor Klančar
Fakulteta za elektrotehniko, UL
Tržaška 25, 1000 Ljubljana
E-pošta: markohancic@gmail.com, ales.sink@siol.net, andrej.zdesar@fe.uni-lj.si, gregor.klancar@fe.uni-lj.si

Localization and control of a mobile system in an environment

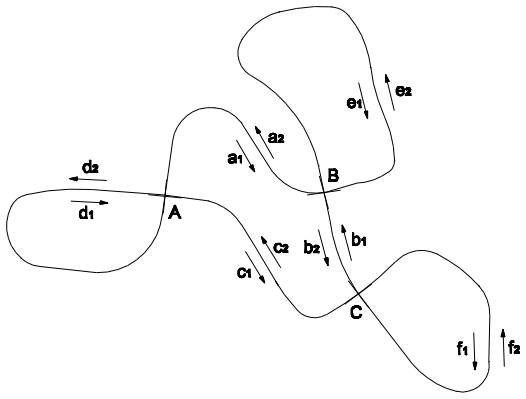
This paper describes line tracking control of a differentially driven mobile system, map building, localization of mobile system using Bayesian filter and navigation of mobile system between crossroads. An industry application for this mobile system could be found in automated warehouses or distribution centers. Mobile system used in this work is a mobile vacuum cleaner iRobot Roomba 521, controlled by a single-board computer (SBC) RaspberryPi.

1 Uvod

Prispevek obravnava vodenje mobilnega sistema z diferencialnim pogonom po črti, gradnjo zemljevida okolja, lokalizacijo mobilnega sistema v okolju in vodenje mobilnega sistema med postajami v zemljevidu.

1.1 Predstavitev problema

Okolje v katerem se mobilni sistem nahaja predstavljajo pravokotna križišča, ki jih povezujejo različno ukrivljene poti. V našem primeru sestavljajo okolje tri križišča A, B in C, ki so med seboj povezana s šestimi potmi. Če upoštevamo, da se po vsaki poti mobilni sistem lahko pelje v dveh smereh, dobimo 12 različnih poti (slika 1).



Slika 1: Zemljevid celotne proge ter označbe poti in križišč. Vidimo, da imamo 12 različnih poti in tri križišča.

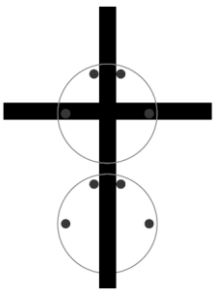
Naloga mobilnega sistema je slediti krivuljam v okolju, zaznavanje križišč in sposobnost izbire smeri v križišču. Prav tako mora biti mobilni sistem na podlagi prej zgrajenega zemljevida okolja sposoben določiti kje v okolju se nahaja[2].

2 Vodenje mobilnega sistema

Vodenje mobilnega sistema je obsegalo sledenje črti v okolju ter zaznavo križišč in izbiro smeri v križišču glede na dan ukaz.

V našem primeru smo kot mobilni sistem uporabili robotski sesalnik iRobot Roomba 521 [3]. Mobilni sistem smo preko serijskega vodila krmilili z uporabo miniračunalnika Raspberry Pi z operacijskim sistemom Linux/Raspbian, na katerem je tekel krmilni program, spisan v jeziku Python 2.6.

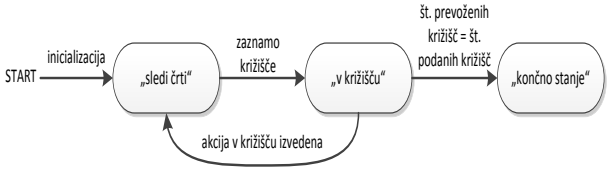
Mobilni sistem Roomba je opremljen z desetimi IR senzorji bližine (šest spredaj in štirje spodaj). Osnovna funkcija spodnjih štirih senzorjev je zaznavanje vrzeli in stopnišč, ki smo jo razširili na zaznavo križišč in črte. Koncept zaznave križišč in črte je predstavljen na sliki 2. S prednjima senzorjema mobilni sistem zaznava črto, s stranskima pa križišča. Z nekaj preizkusi smo določili prag *P*, ki predstavlja mejo med zaznavanjem črte in zaznavanjem okolice.



Slika 2: Koncept zaznavanje črte in križišč z IR senzorji.

Vrednost praga smo nastavili na 1900. Izvedli smo tudi detekcijo napake (odpoved senzorja). V primeru, da je izmerjena vrednost manjša od 200 algoritem vrne vrednost *-1*, kar pomeni napako.

Postopek vodenja mobilnega sistema lahko razdelimo na tri stanja in sicer stanje »*sledi črti*«, stanje »*v križišču*« in »*končno stanje*« (slika 3). Preklopi med stanji se izvedejo, ko algoritem zazna križišče.



Slika 3: Stanja sistema.

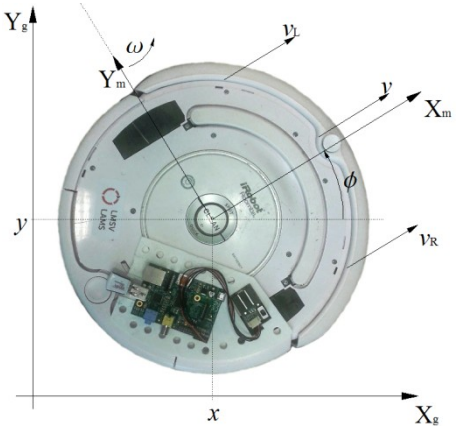
V stanju »*sledi črti*« sistem bere vrednosti sprednjih dveh IR senzorjev. Če noben izmed senzorjev ne zazna črte, se mobilni sistem premika naravnost. V primeru, da kateri od senzorjev zazna črto, se hitrost vrtenja nasprotnega kolesa poveča, hitrost vrtenja soležnega kolesa pa zmanjša za spremembo hitrosti Δv.

Iz stanja »*sledi črti*« v stanje »*v križišču*« sistem preklopi, ko oba stranska senzorja zaznata črto. Stanje »*sledi črti*« je začetno stanje sistema. V »*končno stanje*« sistem preklopi, ko prevozi vnaprej določeno število križišč. V stanju »*v križišču*« se mobilni sistem najprej ustavi, nato preveri kakšno akcijo naj izvede. Možne so štiri akcije: levo, naprej, desno in nazaj. Po izvedeni akciji preidemo nazaj v stanje »*sledi črti*«. Preklop nazaj v stanje »*v križišču*« je zaradi preprečitve napačne zaznave za kratek čas onemogočen.

Ko so izpolnjeni pogoji za preklop v »*končno stanje*« se mobilni sistem ustavi. Neodvisno od stanja v katerem se mobilni sistem nahaja, se izvaja tudi zaznavanje ovire na poti. Oviro mobilni sistem zaznava s sprednjim odbijačem in čelnimi IR senzorji. Ob naletu na oviro se ustavi in ne nadaljuje izvajanja dokler ovira ni odstranjena.

3 Gradnja zemljevida

Gradnjo zemljevida okolja smo izvedli s pomočjo odometrije [1, 5]. Odometrija je postopek, s katerim ocenjujemo spremembo lege mobilnega sistema na podlagi odčitkov meritev senzorjev (inkrementalnih kodirnikov na motorjih).



Slika 4: Kinematika diferencialnega pogona, kjer so ω krožna hitrost, φ kot usmerjenosti mobilnega sistema, v_l obodna hitrost levega kolesa, v_r obodna hitrost desnega kolesa, v tangencialna hitrost mobilnega sistema.

Odometrijo tvorijo vrednosti položaja mobilnega sistema v x in y koordinatah zunanjega koordinatnega sistema (KS) in zasuk glede na zunanji KS ($[x,y,\phi]$).

Odometrijo mobilnega sistema na diferencialni pogon izračunamo s pomočjo kinematičnega modela (slika 4).

Povezavo med kotnimi hitrostmi koles in hitrosti gibanja mobilnega sistema podajata enačbi (1) in (2), kjer so ω krožna hitrost mobilnega sistema in v tangencialna hitrost mobilnega sistema, L razdalja med kolesi, v_r obodna hitrost desnega kolesa, v_l obodna hitrost levega kolesa:

$$\omega(t) = \frac{v_r(t) - v_l(t)}{L} \tag{1}$$

$$v(t) = \frac{v_r(t) + v_l(t)}{2} \tag{2}$$

S pomočjo slike 4 in enačb (1) in (2) lahko ob upoštevanju povezave med zveznim in diskretnim prostorom $t = kT$, kjer je k zaporedni vzorec in T čas vzorčenja, izračunamo lego mobilnega sistema v naslednjem trenutku:

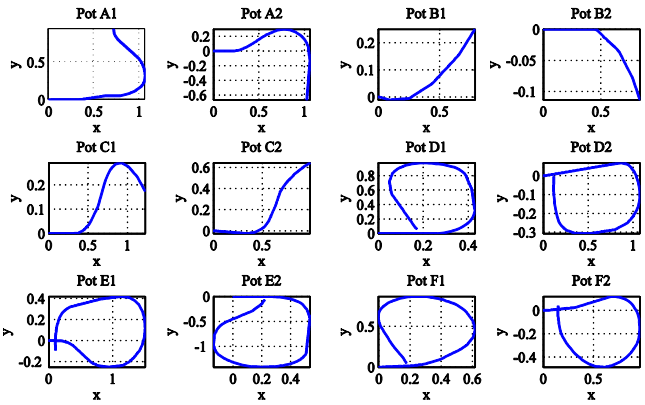
$$x(k+1) = x(k) + v(k) \cdot T \cdot \cos(\varphi(k)) \tag{3}$$

$$y(k+1) = y(k) + v(k) \cdot T \cdot \sin(\varphi(k)) \tag{4}$$

$$\varphi(k+1) = \varphi(k) + \omega(k) \cdot T \tag{5}$$

V zgornjih enačbah (3), (4) in (5) je predpostavljeno, da se ω_l in ω_r med časi vzorčenja ne spreminjata. Numerična integracija enačb (3), (4) in (5) je izvedena z Eulerjevo metodo.

S pomočjo odometrije smo izračunali koordinate točk vseh poti med križišči. Zemljevid smo zgradili po postopku, ki sledi. Ko je mobilni sistem prispel do križišča, smo nastavili začetno vrednost koordinat, nato smo mobilni sistem napotili proti naslednjemu križišču in ob vsakem času vzorčenja shranili izračunano trenutno lego mobilnega sistema. Ko je mobilni sistem prispel do naslednjega križišča, se je ustavil in posnete podatke o prevoženi poti shranil v datoteko CSV (Comma Separated Value).



Slika 5: Poti, ki sestavljajo zemljevid. Za vsako pot sta prikazani meritvi poti v obe smeri.

Tak postopek smo ponovili za vse poti v okolju (slika 1). Dobljene posnete podatke smo uvozili in izrisali v okolju Matlab (slika 5). Vse koordinate na sliki so podane v metrih.

4 Lokalizacija mobilnega sistema v okolju

Mobilni sistem je moral po prihodu v križišče ugotoviti kje v okolju se nahaja. Mobilni sistem je ob zapustitvi križišča ponastavil meritve odometrije in med potovanjem po poti proti naslednjemu križišču shranjeval podatke iz odometrije. Ob prispetju do naslednjega križišča, je primerjal prevoženo pot s shranjenimi potmi in vrnil oceno položaja.

Evklidsko razdaljo smo uporabili kot kriterij primerjave med prevoženo in shranjenimi potmi:

$$J = \sum_{i=1}^{N} \|z(i) - z_m(i)\|, \qquad (6)$$

kjer J predstavlja vrednost kriterija, N število vzorcev, i zaporedni vzorec, $z(i)$ točko poti iz zemljevida in $z_m(i)$ točka izmerjene poti. Za lažjo primerjavo smo vse izmerjene poti prevzorčili na 50 vzorcev.

Lokalizacijo mobilnega sistema v danem okolju smo izvedli z uporabo Bayesovega filtra. Bayesov filter je sestavljen iz dveh korakov: predikcijskega in korekcijskega koraka [1, 2 in 4]. Pri predikcijskem koraku upoštevamo poznano trenutno akcijo in tako lahko na podlagi modela sistema napovemo verjetnostno porazdelitev stanj. V našem primeru obravnavamo diskretni primer zato bomo podali Bayesov filter v diskretni obliki. Izračun predikcije opravimo po enačbi (77), kjer $p(x_k|z_{1:k-1}, u_{1:k})$ predstavlja predikcijo porazdelitve verjetnosti stanj na osnovi preteklih meritev (do k-1) in opravljenih akcij do trenutka k. Verjetnost $p(x_k|x_{k-1}, u_k)$ je porazdelitev verjetnosti za prehajanja med stanji, v našem primeru je to zanesljivost delovanja izbire smeri v križišču. Verjetnost $p(x_{k-1}|z_{1:k-1}, u_{1:k-1})$ je korekcijska ocena porazdelitve verjetnosti prejšnjega stanja.

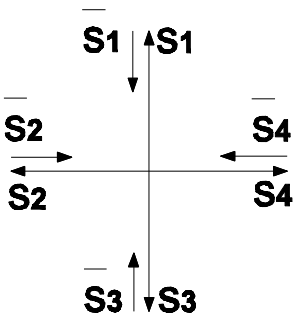
$$p(x_k|z_{1:k-1}, u_{1:k}) = \sum_{x_{k-1} \in P \cup \bar{P}} p(x_k|x_{k-1}, u_k) p(x_{k-1}|z_{1:k-1}, u_{1:k-1}) \qquad (7)$$

Pri korekcijskem koraku lahko s pomočjo meritev senzorjev popravimo s predikcijo ocenjeno informacijo o legi mobilnega sistema v okolju. Izračun korekcije izvedemo z enačbo (8), pri čemer je $p(z_k|x_k)$ verjetnost meritve glede na stanje, $p(x_k|z_{1:k-1}, u_{1:k})$ predikcija ocene porazdelitve verjetnosti stanja izračunana v predikcijskem koraku in η normirni faktor (9).

$$p(x_k|z_{1:k}, u_{1:k}) = \eta p(z_k|x_k) p(x_k|z_{1:k-1}, u_{1:k}) \qquad (8)$$

$$\eta = \frac{1}{\sum_{x_k \in P \cup \bar{P}} p(z_k|x_k) p(x_k|z_{1:k-1}, u_{1:k})} \qquad (9)$$

Ker izbira smeri ni vedno enoznačna, smo za naš primer za vsako akcijo U_k (naprej, nazaj, levo, desno) določili verjetnosti, da bo katera izmed smeri izbrana glede na izbrano akcijo in smerjo vstopa v križišče, za vse možne smeri (slika 6). V tabeli 1 so zbrane verjetnosti za izstop v smeri s_1 pri vstopih v vseh nasprotnih smereh glede na izbrane štiri možne akcije U_k. Vrednosti so identične za druge tri vstope v križišče ob upoštevanju zamenjav oznak izstopov in vstopov.



Slika 6: Oznaka smeri, ki vstopajo (s prečno črto) in izstopajo iz križišča.

Za potrebe računanja verjetnosti in lokalizacije smo sestavili tabelo obratnih preklopov med potmi (tabela 2). Tabela nam podaja možne smeri poti po katerih smo vstopali v križišče (slika 6), če želimo v križišču pot nadaljevati v smeri s_1 glede na zemljevid poti (slika 1).

Tabela 1: Zanesljivost delovanja izbire smeri v križišču.

verjetnost \ akcija - U_k	naprej	nazaj	levo	desno
$p(X_k = s_1 \| X_{k-1} = \bar{s_1}, U_k)$	0,05	0,70	0,10	0,10
$p(X_k = s_1 \| X_{k-1} = \bar{s_2}, U_k)$	0,10	0,10	0,70	0,05
$p(X_k = s_1 \| X_{k-1} = \bar{s_3}, U_k)$	0,75	0,10	0,15	0,15
$p(X_k = s_1 \| X_{k-1} = \bar{s_4}, U_k)$	0,10	0,10	0,05	0,70

Za preizkus delovanja lokalizacije z Bayesovim filtrom smo določili zaporedje ukazov akcij in sledečih poti. Na začetku smo za vse poti predpostavili enako verjetnost, da se mobilni sistem nahaja na njej oziroma jo je prevozil. Mobilni sistem je začel svoje potovanje in meril prvo pot do prihoda v prvo križišče. Shranjene meritve poti smo primerjali s prevoženo izmerjeno potjo in tako glede na izbran kriterij (6) določili verjetnosti za vse možne prevožene poti. Dobljene verjetnosti smo uporabili za korekcijski korak Bayesovega filtra.

V predikcijskem koraku smo tako ob upoštevanju začetne porazdelitve verjetnosti in izbrane akcije ter zanesljivosti izbire smeri izračunali predikcijo. S pomočjo meritev smo to napoved popravili in izračunali v katerem križišču se najbolj verjetno nahajamo.

Verjetnost, v katerem križišču se nahajamo, smo izračunali kot vsoto verjetnosti poti, ki sestavljajo križišče. Izračunane verjetnosti smo uporabili v naslednjem predikcijskem koraku in napoved zopet

popravili s korekcijo, tako smo celoten postopek ponavljali.

Tabela 2: Obratni preklopi med potmi.

s_1	*a1 b1 c1 d1 e1 f1 a2 b2 c2 d2 e2 f2*
$\bar{s_1}$	*a2 b2 c2 d2 e2 f2 a1 b1 c1 d1 e1 f1*
$\bar{s_2}$	*d1 c1 a2 c2 a1 b2 b1 e1 f1 d2 e2 f2*
$\bar{s_3}$	*d2 f1 d1 a2 b1 c1 e1 e2 f2 c2 a1 b2*
$\bar{s_4}$	*c2 f2 d2 d1 e1 f1 e2 a1 b2 a2 b1 c1*

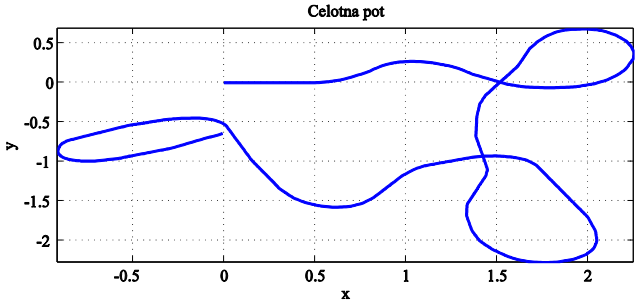
V tabeli 3 je prikazano izbrano zaporedje akcij mobilnega sistema, predvidene prevožene poti in križišča ter najbolj verjetne poti in križišča ugotovljene s pomočjo Bayesovega filtra. Vidimo lahko, da je Bayesov filter v večini primerov z veliko verjetnostjo (nad 0,86) pravilno ugotovil lego mobilnega sistema.

Tabela 3: Izbrano zaporedje merjenih poti, akcij (1-naprej, 2-nazaj, 3-levo, 4-desno) in dejanskih prevoženih križišč ter z uporabo Bayesovega filtra locirane poti in križišča, s pripadajočimi vrednostmi verjetnosti.

Izmerjene poti	*d1*	*c1*	*f2*	*b1*	*e1*	*b2*	*c2*	*a1*	*e2*	*a2*	*d1*	*d2*
Akcija	1	4	4	4	1	3	4	4	1	1	1	2
Križišča	*A*	*C*	*C*	*B*	*B*	*C*	*A*	*B*	*B*	*A*	*A*	*A*
Locirane poti	*d1*	*c1*	*f2*	*b1*	*e1*	*b2*	*c2*	*a1*	*e2*	*a2*	*d1*	*d2*
Verjetnost	0,53	0,86	0,97	0,96	0,98	0,98	0,97	0,97	0,99	0,61	0,94	0,96
Locirana križišča	*A*	*C*	*C*	*B*	*B*	*C*	*A*	*B*	*B*	*A*	*A*	*A*
Verjetnost	0,62	0,88	0,98	0,96	0,99	0,98	0,97	0,97	0,99	0,61	0,95	0,97

5 Vodenje mobilnega sistema med postajami v zemljevidu

Predpostavimo primer, da imamo mobilni sistem v skladišču, ki mora pripeljati material iz skladišča do delovne postaje. Pri tem poznamo zemljevid okolja. Ko mobilni sistem dobi nalogo, mu povemo skozi katera križišča se mora peljati in kakšno akcijo (obrat) mora izvesti v križiščih. Mobilnemu sistemu smo podali zaporedje akcij za doseg želene delovne postaje. Podali smo mu zaporedje šestih enakih akcij za vožnjo naravnost. V tem primeru je moral mobilni sistem prevoziti vse poti v zemljevidu in se vrniti nazaj v začetno točko. Zemljevid prevožene poti, izmerjen s pomočjo odometrije, je prikazan na sliki 7.



Slika 7: Zemljevid, izmerjen s pomočjo odometrije.

6 Zaključek

Največji problem pri vodenju mobilnega sistema je bilo zaznavanje križišč. Mobilni sistem je križišča zaznaval brez napak, če so bile črte nepoškodovane in čiste ter so se poti sekale pravokotno. Z uporabo naprednejšega algoritma za detekcijo križišč ali z večjim številom senzorjev, bi se lahko povečalo robustnost sistema. Zaradi netočnega modela mobilnega sistema in zdrsov pogonskih koles se napaki pri računanju odometrije ne moremo v celoti izogniti in znaša po celotni prevoženi poti 0,6 m (slika 7).

Možnost za razširitev sistema je v sistemu vodenja med postajami v zemljevidu. Lokalizacija s pomočjo Bayesovega filtra se je za naš primer izkazala za uporaben način ugotavljanja lege mobilnega sistema. Sistem vodenja bi lahko nadgradili tako, da bi se mobilni sistem najprej lokaliziral, nato pa izračunal najbližjo pot do želenega križišča.

Primer lokalizacije opisan v tem prispevku prinaša mobilnemu sistemu večjo avtonomijo delovanja. Tak mobilni sistem bi lahko opravljal različne naloge v avtomatiziranih skladiščih in razdelitvenih središčih.

7 Literatura

[1] G. Klančar: Avtonomni mobilni sistemi, Fakulteta za elektrotehniko v Ljubljani, Ljubljana 2013, Slovenija;

[2] G. Klančar, A. Zdešar.: Avtonomni mobilni sistemi - Navodila za laboratorijske vaje: Vodenje mobilnega sistema Roomba, 2013, Fakulteta za elektrotehniko v Ljubljani, Ljubljana 2013, Slovenija

[3] iRobot® Roomba 500 Open Interface (OI) Specification, http://www.irobot.lv/uploaded_files/File/iRobot_Roomba_500_Open_Interface_Spec.pdf

[4] S. Thrun, W. Burgard, D. Fox: Probabilistic Robotics, The MIT Press, 2006

[5] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun, Principles of Robot motion, Theory, Algorithms and Implementation, MIT Press, 2004