

Lokalizacija kolesnega mobilnega sistema na osnovi zaznavanja posebnih značk s stereo kamero

Martin Anton Škoberne¹, as. dr. Andrej Zdešar¹

¹Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška cesta 25, 1000 Ljubljana
E-pošta: ms1843@student.uni-lj.si, andrej.zdesar@fe.uni-lj.si

Localisation of the wheeled mobile system based on detection of special markers with a stereo camera

The purpose of the work described in this article is localization of the wheeled mobile system. The robot is equipped with a stereo camera that enables detection of special markers. The global positions of the markers are known. The robot measures the distance vector from the stereo camera to the visible marker. To determine the position of the robot, it must detect at least two markers at the same time. Based on the measured data and the odometry, the estimate of the robot pose can be calculated. We implement this with a localization algorithm. In the presented article two localization algorithms are implemented, namely: Extended Kalman Filter (EKF) and Particle Filter (PF). The solution was developed in the Robot Operating System (ROS) environment.

1 Uvod

Avtonomni mobilni sistemi imajo mnogo področij uporabe. Zmožni so samostojnega premikanja v okolju, kjer je poudarek na samostojnih odločitvah ter izvajanju akcij. Za to morajo biti opremljeni s primernimi senzorji. Kamera predstavlja vse bolj popularen senzor za zaznavanje oziroma prepoznavanje objektov. Razširjena je uporaba stereo kamere, saj se lahko s primernimi algoritmi oceni tudi globina objektov. Na tem področju obstaja že mnogo rešitev. V [1], so avtorji implementirali algoritme, ki temeljijo na strojnem vidu, za detekcijo, prostorsko sledenje in ocenjevanje 3D-položaja poznanih tarč (značke Aruco) s stereo kamero na enoti za zasuk in nagib.

Da se lahko avtonomni mobilni sistemi avtonomno premikajo po zelenih poteh, morajo poznati svojo lego v prostoru. Tu nastopijo algoritmi za lokalizacijo, zaradi katerih je mogoče avtonomno delovanje mobilnih sistemov. Po prostoru je možno razporediti posebne značke, avtonomni mobilni sistem pa se lahko glede na le-te orientira. Značke zaznava s stereo kamero in glede na lokalni koordinatni sistem oceni razdaljo in kot do njih. Z združevanjem informacij iz odometrije s predhodnimi ocenami lege in meritvami (izračun lege glede na položaje značk) je možno izboljšati oceno lege robota v globalnem koordinatnem sistemu. Za ta namen se običajno uporabljajo pristopi lokalizacije, ki temeljijo na uporabi ene

izmed različic Bayesovega filtra. V tem članku sta predstavljeni dve različici, ki smo ju implementirali v simulaciji, to sta razširjeni Kalmanov filter (EKF) in filter delcev (PF).

V poglavju 2 je predstavljen kolesni mobilnega sistema in njegov model. V poglavju 3 je predstavljena meritev lege robota na podlagi meritev 3D-položaja značk. Lokalizacija je predstavljena v poglavju 4, kjer sta opisana algoritma EKF in PF. Rezultati simulacije in validacije algoritmov so predstavljeni v poglavju 5.

2 Model kolesnega mobilnega sistema in kamere

2.1 Položaj kamere glede na robota

Kolesni mobilni sistem, ki smo ga uporabili za implementacijo tega dela, je štirikolesni mobilni robot Pioneer 3-AT. Na robota je nameščena stereo kamera. Predpostavili smo, da lahko jemljemo stereo kamero kot senzor, ki meri 3D-položaj poznanih tarč. Zato definiramo le en koordinatni sistem za stereo kamero, katerega namen je preslikava meritev. Definiramo statično preslikavo med koordinatnim sistemom robota R in koordinatnim sistemom kamere C z rotacijsko in translacijsko matriko

$$\mathbf{R}_C^R = \text{Rot}_z(-\frac{\pi}{2})\text{Rot}_x(-\frac{\pi}{2}) = \begin{bmatrix} 0 & 0 & 1 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \quad (1)$$

$$\mathbf{t}_C^R = [0,4 \quad 0 \quad 0,6]^T.$$

Transformacija (1) je prikazna na sliki 2, kjer je v simulacijskem orodju Rviz prikazana lega koordinatnega sistema kamere glede na koordinatni sistem robota (poglavje 5).

2.2 Model robota

Za modeliranje gibanja robota smo uporabili model diferencialnega pogona. Izpeljava in dodatna razlaga razvoja modela z diferencialnim pogojem je podana v [3]. Zunanja kinematika v diskretni obliki je v globalnih koordinatah za model z diferencialnim pogonom podana kot

$$\begin{aligned} x(k+1) &= x(k) + \Delta d(k) \cos \varphi(k) & \Delta d(k) &= T_s v(k) \\ y(k+1) &= y(k) + \Delta d(k) \sin \varphi(k) & \Delta \varphi(k) &= T_s \omega(k) \\ \varphi(k+1) &= \varphi(k) + \Delta \varphi(k) \end{aligned} \quad (2)$$

Vektor $\mathbf{x} = (x, y, \varphi)$ označuje stanje sistema – to je lega robota v globalnem koordinatnem sistemu. Koordi-

nati x in y označujeta pozicijo v globalnem koordinatnem sistemu, φ pa zasuk. Vhodni veličini v sistem sta translatorsna hitrost robota v v smeri x -osi robota in kotna hitrost ω okoli z -osi robota. Z oznako k označimo diskretni trenutek, kjer je čas vzorčenja označen z oznako T_s . Oznaki Δd in $\Delta\varphi$ sta spremembi premika v smeri osi x in orientacije okoli osi z .

Lego robota v kateremkoli časovnem trenutku dobimo z integracijo kinematičnega modela (2), kar imenujemo odometrija. Pri realnem kolesnem mobilnem sistemu prihaja do napak pri izračunu odometrije, zato smo v simulacijo vključili šum. Za model šuma smo uporabili Gaussov šum, ki ga označimo z $N(\mu, \sigma^2)$, kjer je μ srednja vrednost in σ^2 varianca. Pri izračunu odometrije smo šum vključili tako, da smo Δd in $\Delta\varphi$ pomnožili z Gaussovim šumom

$$\begin{aligned}\Delta d_N(k) &= \Delta d(k)N(0, 0.002) \\ \Delta\varphi_N(k) &= \Delta\varphi(k)N(0, 0.01).\end{aligned}\quad (3)$$

2.3 Sistem za zaznavanje lege posebnih značk

Modelu robota je sledila izvedba sistema za zaznavanje lege posebnih značk. To je simulacija delovanja stereo kamere za ocenjevanje 3D-položaja poznanih tarč glede na koordinatni sistem stereo kamere. Za pravilno delovanje simulacije smo meritvam prišteli šum, ki narašča z oddaljenostjo od značke. Šum smo modelirali kot Gaussov šum, kjer je standardna deviacija premo sorazmerna z absolutno vrednostjo razdalje v smeri z -osi kamere. Meritev z_c modeliramo kot

$$\mathbf{z}_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} + |z_c| \begin{bmatrix} N(0, 0.001) \\ N(0, 0.001) \\ N(0, 0.005) \end{bmatrix} \quad (4)$$

kjer je vektor (x_c, y_c, z_c) pravilni 3D-položaj značke glede na koordinatni sistem stereo kamere.

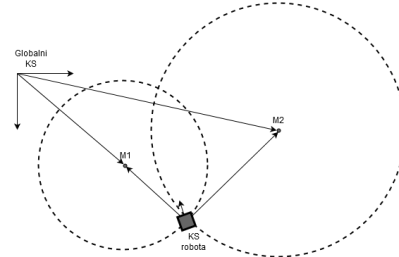
Za določevanje položaja značk glede na robota, smo morali meritev preslikati iz koordinatnega sistema kamere v koordinatni sistem robota. To je statična transformacija, ki je definirana z rotacijsko in transformacijsko matriko (1). Pri robotu velja tudi posebna omejitev, to je zorni kot. Torej s kamero ne moremo zaznati značk, ki se jih ne vidi.

3 Meritev lege

Za meritev lege je potrebno izračunati oceno lege robota glede na lege posebnih značk, ki jih robot lahko zazna. Lego definira vektor (x, y, φ) . Koordinato z lahko zanemarimo, saj se robot giblje po tleh. Najprej je treba vektorje razdalj transformirati iz koordinatnega sistema kamere v koordinatni sistem robota. Transformiramo jih s pomočjo rotacijskega in translatorsnega premika (1). Tako dobimo vektorje razdalj od koordinatnega sistema robota do značk.

Vsaka značka je unikatna, tako da jih lahko ločimo med sabo. Za izračun lege sta potrebni vsaj dve vidni znački. Lege značk v globalnem koordinatnem sistemu so znane. Torej imamo na razpolago: pozicije značk v globalnem koordinatnem sistemu in vektorje razdalj, ki ocenjujejo pozicije značk glede na koordinatni sistem robota. Z informacijo o poziciji značk v globalnem koor-

dinatnem sistemu in absolutni razdalji med robotom in značkami izračunamo presek dveh krožnic (slika 1). Za posamezno krožnico si lahko predstavljamo, da je njen center v koordinatah značke, radij krožnice je pa enak absolutni razdalji robota do značke. Tako dobimo dve presečiščici – to sta dve možni legi.



Slika 1: Meritev lege robota.

Za izbiro pravilne lege si pomagamo z vektorji razdalj od robota do značk. To storimo tako, da si pri obeh potencialnih legah izberemo enega od vektorjev razdalj do značke in preverimo ujemanje drugega. Izračunamo absolutno razliko med dejansko lego izbrane značke in obema izračunanimi legama za to značko. Pri tisti potencialni legi, kjer je absolutna razlika manjša, pomeni, da je le-ta prava lega.

Če ne bi imeli unikatnih značk, ki jih kamera razpozna oz. bi izmerili lahko le absolutno razdaljo do značk, bi za pravilno določitev lege potrebovali meritve razdalj vsaj treh značk namesto dveh.

4 Lokalizacija

Da se lahko avtonomni mobilni sistemi avtonomno gibljejo po želenih poteh, morajo poznati svojo lego v prostoru. V našem primeru si lahko za ocenitev lege pomagamo z naslednjimi podatki: odometrija na osnovi računanja poti robota glede na njegov premik in meritve na osnovi zaznavanja značk.

Sama odometrija nam ne daje dovolj dobre ocene lege, saj ne moremo zagotoviti lege robota v neznanem okolju. Poleg tega je odometrija zaradi integracije podvržena lezenju. Meritve na osnovi zaznavanja značk pa niso tako pogoste, kot na primer podatki odometrije. Poleg tega tudi niso vedno na voljo – robot jih ne zazna, če so izven dosega. Meritve opravljene s kamero moramo v postopku lokalizacije primerno združiti s predhodnimi podatki o legi mobilnega sistema. Tukaj uporabimo algoritme, ki temeljijo na uporabi ene izmed različic Bayesovega filtra. V tem delu smo se osredotočili na razširjeni Kalmanov filter (EKF) in filter delcev (PF), ki so opisani v [3, 4, 5].

4.1 Razširjeni Kalmanov filter

Kalmanov filter je razvit za linearne sisteme, vsi šumi morajo biti izraženi z normalno porazdelitvijo. Naš model je nelinearen, zato uporabimo razširjeni Kalmanov filter (EKF), kjer se nelinearnost aproksimira z linearnim modelom. Z linearizacijo dobimo občutljivostne matrike za trenutne vrednosti ocenjenih stanj in meritev.

Splošen nelinearni model sistema, kjer modeliramo tudi šum, je podan kot

$$\begin{aligned}\mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k + \mathbf{v}_k).\end{aligned}\quad (5)$$

Stanje sistema $\mathbf{x}_k$ je podano kot funkcija $\mathbf{f}$, ki je odvisna od prejšnjega stanja $\mathbf{x}_{k-1}$, vhoda $\mathbf{u}_{k-1}$ in šuma $\mathbf{w}_{k-1}$, ki lahko nastopa na vhodu sistema ali pa vpliva direktno na stanja sistema. V našem primeru je stanje sistema lega robota, vhod sta translatorska in kotna hitrost, šum je pa posledica napake zaznavanja pravega premika robota zaradi netočnosti enkoderjev, trenja in zdrsa koles ali drugih pojavov. Model meritve $\mathbf{z}_k$ je podan kot funkcija $\mathbf{h}$, ki modelira senzor. V našem primeru je ta model opisan v poglavju 3, kjer je razloženo kako pridemo do izmerjene lege robota glede na zaznane značke. Tudi pri modelu meritve je potrebno določiti šum $\mathbf{v}_k$, ki se prišteje h končni oceni meritve.

EKF je podan s predikcijskim korakom

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}) \quad (6)$$

in korekcijskim korakom

$$\mathbf{P}_{k|k-1} = \mathbf{A}\mathbf{P}_{k-1|k-1}\mathbf{A}^T + \mathbf{F}\mathbf{Q}_{k-1}\mathbf{F}^T$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{C}^T(\mathbf{C}\mathbf{P}_{k|k-1}\mathbf{C}^T + \mathbf{R}_k)^{-1}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{x}}_k) \quad (7)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{C}\mathbf{P}_{k|k-1}$$

V predikcijskem koraku uporabimo nelinearni model za oceno predikcije stanja. Korekcijo pa izvedemo, ko prejmemo novo meritev in s tem popravimo oceno predikcije. Nelinearni model mora biti lineariziran okoli trenutno ocenjenega stanja.

Nelinearni model za kolesni mobilni sistem, kjer nastopa tudi šum, predpostavimo, da je

$$\mathbf{x}_k = \begin{bmatrix} x_{k-1} + (\Delta d_{k-1} + w_{\Delta d}) \cos \varphi_{k-1} \\ y_{k-1} + (\Delta d_{k-1} + w_{\Delta d}) \sin \varphi_{k-1} \\ \varphi_{k-1} + (\Delta \varphi_{k-1} + w_{\Delta \varphi}) \end{bmatrix}, \quad (8)$$

kovariančni matriki šumov $\mathbf{Q}$ in $\mathbf{R}$ sta podani kot

$$\begin{aligned} \mathbf{Q} &= E[\mathbf{w}\mathbf{w}^T] \quad \mathbf{R} = E[\mathbf{v}\mathbf{v}^T] \\ \mathbf{w} &= \begin{bmatrix} w_{\Delta d} \\ w_{\Delta \varphi} \end{bmatrix} = \begin{bmatrix} N(0, 0.002) \\ N(0, 0.01) \end{bmatrix} \\ \mathbf{v} &= \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} N(0, 0.001) \\ N(0, 0.001) \\ N(0, 0.005) \end{bmatrix}, \end{aligned} \quad (9)$$

kjer vektor $\mathbf{w}$ modelira šum ob premiku robota, vektor $\mathbf{v}$ pa šum meritve. Obe kovariančni matriki sta diagonalni, saj smo predpostavili, da šumi niso med seboj korelirani. Načeloma sta modela šumov odvisna tudi od hitrosti gibanja robota in od oddaljenosti izmerjenih razdalj do značk, vendar smo v simulaciji dosegli dobre rezultate s statičnimi vrednostmi kovariančnih matrik. Definiramo Jacobijevo matriko $\mathbf{A}$ za prehajanje šuma iz prejšnjih vrednosti stanja

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & -\Delta d_{k-1} \sin \varphi_{k-1} \\ 0 & 1 & \Delta d_{k-1} \cos \varphi_{k-1} \\ 0 & 0 & 1 \end{bmatrix} \quad (10)$$

in matriko $\mathbf{F}$ za vpliv šuma iz vhodov na stanja

$$\mathbf{F} = \begin{bmatrix} \cos \varphi_{k-1} & 0 \\ \sin \varphi_{k-1} & 0 \\ 0 & 1 \end{bmatrix}. \quad (11)$$

Matrika $\mathbf{C}$ opisuje vpliv šuma iz stanja na meritve. V našem primeru je meritev v takšni obliki kot stanje sistema, zato je $\mathbf{C} = \mathbf{I}_{3 \times 3}$.

4.2 Filter delcev

Pri Kalmanovem filtru je potrebno predpostaviti Gaussove porazdelitve verjetnosti in linearnost sistema. V primeru nelinearnih sistemov pa lahko nelinearnost modela lineariziramo, kar nas privede do EKF. Filter delcev je bolj splošen pristop, kjer ne zahtevamo Gaussove porazdelitve in linearnosti sistema. Osnovna ideja je, da trenutno oceno porazdelitve verjetnosti po opravljeni meritvi aproksimiramo z množico delcev. Tako vsak delec predstavlja svojo oceno o dejanskem stanju sistema. Opis porazdelitve verjetnosti s pomočjo naključno generiranih delcev predstavlja neparametričen opis porazdelitve verjetnosti, ki ni omejena le na Gaussovo porazdelitev. Nadalje omogoča tudi modeliranje nelinearnih transformacij šuma.

Algoritem filtra delcev (PF) opisujejo naslednji koraki. Najprej se izvede inicializacija z množico N delcev $\mathbf{x}_k^i$, ki se jih postavi na naključne začetne vrednosti glede na porazdelitev $p(\mathbf{x}_0)$. Določimo naključno normalno porazdelitev, ki jo prištejemo začetnim stanjem delcev. Pri inicializaciji še ne moremo oceniti pomembnosti delcev, zato imajo vsi delci enako utež. Sledi predikcija, kjer za vsak delec izvedemo premik z modelom premika in znanim vhodom, kateremu dodamo naključno vrednost glede na šumne lastnosti. Za vsak delec $\mathbf{x}_k^i$ izračunamo oceno novega stanja

$$\hat{\mathbf{x}}_k^i = \begin{bmatrix} x_{k-1}^i + (\Delta d_{k-1} + w_{\Delta d}^i) \cos \varphi_{k-1}^i \\ y_{k-1}^i + (\Delta d_{k-1} + w_{\Delta d}^i) \sin \varphi_{k-1}^i \\ \varphi_{k-1}^i + (\Delta \varphi_{k-1} + w_{\Delta \varphi}^i) \end{bmatrix}, \quad (12)$$

kjer šum modeliramo kakor v (9). Ko prejmemo novo meritev, izvedemo korekcijo. Pri korekciji za vsak delec ocenimo vrednost meritve, ki bi jo sistem izmeril, če bi njegovo stanje ustrezalo stanju delca. Glede na dejansko izmerjeno meritev in primerjavo z ocenjenimi meritvami delcev ovrednotimo pomembnost delcev. Odstopanje dejanske in ocenjene meritve imenujemo inovacija

$$innov_k^i = \mathbf{z}_k^i - \hat{\mathbf{z}}_k^i, \quad (13)$$

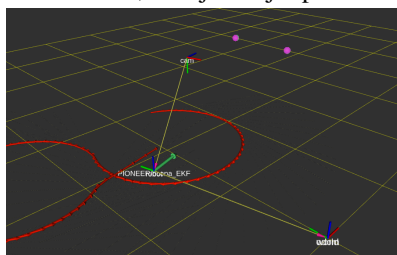
kjer imajo bolj verjetni delci manjše odstopanje. Za vsak delec ocenimo njegovo pomembnost oziroma uteži w_k^i , ki jih določimo z Gaussovo porazdelitvijo verjetnosti

$$w_k^i = \det(2\pi\mathbf{R})^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{innov}_k^i)^T \mathbf{R}^{-1}(\mathbf{innov}_k^i)}. \quad (14)$$

Nov nabor delcev se določi glede na njihovo pomembnost. Iz nabora delcev se naključno izbere delce z verjetnostjo proporcionalno njihovi pomembnosti. Tako so bolj pomembni delci izbrani večkrat, manj pomembni delci pa manjkrat. Postopek izbora nove generacije delcev izvedemo tako, da uteži delcev normiramo z vsoto uteži ter jih kumulativno seštejemo. Delci z večjo verjetnostjo zavzemajo na intervalu od 0 do 1 večje področje kot manj verjetni. Naključno izberemo N števil od 0 do 1 in pogledamo katerim delcem se prilagajajo, izbrane delce uporabimo v korekcijskem koraku. Ocena stanja filtra je nato enaka povprečni vrednosti stanja delcev, za kot zasuk φ smo izbrali vrednost delca z največjo verjetnostjo. Te korake izvajamo v vsakem časovnem trenutku s trenutnimi vhodi in meritvami ter predhodno oceno stanja.

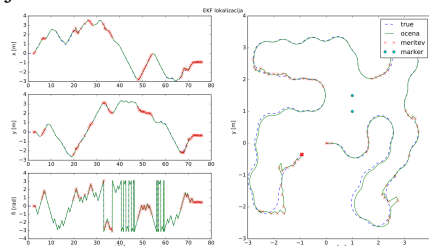
5 Rezultati

Vso delo, predstavljeno v tem članku, smo izvedli v robotskem operacijskem sistemu ROS (angl. *Robot Operating System*) [2]. Za izvedbo simulacije smo v okolju ROS napisali *Python* skripte za simulacijo in uporabili orodje *Rviz* za vizualizacijo (slika 2). Določili smo model robota in sistem za zaznavanje značk, ki modelira meritve 3D položaja značk glede na kamero robota. Te meritve smo preslikali v koordinatni sistem robota in jih vključili v algoritem za lokalizacijo. Robota smo upravljali sami, kjer smo kot vhod podali translatorsno hitrost v smeri x -osi in kotno hitrost okoli z -osi robota. Robota smo poljubno peljali po prostoru tako, da je nekaj časa lahko zaznaval značke, nekaj časa jih pa ni videl.

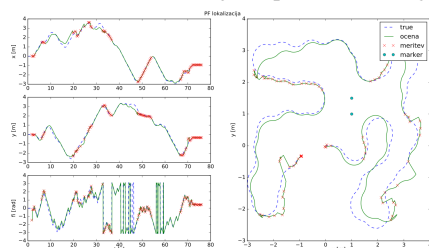


Slika 2: Primer simulacije v *Rviz* - prikaz robota s kamero, značk in oceno lege, ki jo pridobimo z algoritmom lokalizacije.

Rezultati simulacije so ponazorjeni na slikah 3 in 4, kjer lahko na levi strani vidimo potek stanj v odvisnosti od časa, na desni strani pa pot v xy -ravnini, ki jo je robot opravil, označeni sta tudi znački. Vsi grafi imajo za primerjavo podano še pravo pot robota, kjer lahko primerjamo odstopanje izračunane poti, ki smo jo dobili z algoritmom lokalizacije. S križci je označeno, kdaj je bila na voljo meritve in tako opravljena korekcija. Opazimo, da se napaka povečuje, če opravljamo samo predikcijo brez korekcije.



Slika 3: Rezultat lokalizacije z uporabo EKF-algoritma.



Slika 4: Rezultat lokalizacije z uporabo PF-algoritma.

Že iz grafov se vidi, da smo dosegli nekoliko boljše rezultate z algoritmom EKF. To potrdimo tudi z izračunano povprečno srednjo napako (MSE). Pri poskusu prikazanem na sliki je vrednost MSE za algoritem EKF enaka 0,0071 in za algoritem PF je enaka 0,1172. Opazimo, da EKF dosega dosti boljši rezultat, vendar sta kljub temu oba rezultata zelo dobra.

6 Zaključek

V tem članku smo se spoznali z algoritmi lokalizacije za kolesni mobilni sistem. Najprej smo morali ustvariti model robota in simulacijsko okolje, da smo lahko implementirali algoritme za lokalizacijo. Za to smo uporabili ROS-okolje, skripte smo pisali v programskem jeziku *Python*. Za lokalizacijo smo implementirali in primerjali dva algoritma, to sta EKF in PF. Pri obeh smo združevali podatke prejšnjega stanja modela z odometrijo in meritvijo lege robota. Oba algoritma temeljita na dveh korakih, to sta predikcija in korekcija. Predikcija se izvede ob vsaki spremembi stanja modela, korekcija pa le, ko je na voljo nova meritev.

EKF je računsko učinkovit algoritem zaradi linearizacije nelinearnih modelov ter aproksimacije šumov, ki nimajo Gaussove porazdelitve. Zato je pogosto uporabljen v praksi. Točnost, ki jo z linearno aproksimacijo dosežemo, je odvisna od velikosti variance šuma, ter od stopnje nelinearnosti. Zavedati se moramo, da lahko zaradi napake linearizacije filter slabše konvergira k pravi rešitvi. EKF bi bil verjetno bolj primeren za računanje v realnem času. PF je računsko bolj zahteven algoritem, sploh če uporabljamo veliko število delcev za aproksimacijo modela in kjer imamo opravka z večjimi dimenzijami. PF omogoča opis nelinearnih sistemov in poljubne porazdelitve šuma. Prednosti so robustnost in zmožnost rešitve problema globalne lokalizacije in problema ugrabljenega robota.

V našem primeru kažejo rezultati v prid implementaciji EKF-algoritma. To je mogoče tudi zato, ker dobro poznamo model robota in lahko izračunamo prehodne matrike. Če temu ne bi bilo tako, bi mogoče PF prišel bolj prav. Oba algoritma sta dosegla dobre rezultate. Veliko boljše rezultate bi dosegli, če bi v vsakem koraku s predikcijo lahko izvedli tudi korekcijo – tako so bili tudi zasnovani ti algoritmi. Tako se pa nedoločenost ob vsakem koraku predikcije večja, kar se nekoliko bolj pozna pri PF.

Delo bi lahko nadaljevali tako, da bi implementirane algoritme testirali na realnem robotu. ROS daje dobro podporo za komuniciranje med napravami in ponovno uporabo že spisanih funkcij. Boljše rezultate bi lahko tudi dosegli, če bi v prostor postavili več značk in tako dosegli natančnejše meritve.

Literatura

- [1] T. Belcijan, M. Vačovnik in A. Zdešar, Sledenje in ocenjevanje 3D položaja tarč s stereo kamero na enoti za zasuk in nagib. Zbornik desete konference Avtomatizacija v industriji in gospodarstvu, 6. in 7. april 2017, Maribor, Slovenija
- [2] ROS. <http://wiki.ros.org/>
- [3] G. Klančar, A. Zdešar, S. Blažič in I. Škrjanc, Wheeled mobile robotics: from fundamentals towards autonomous systems. Elsevier, Butterworth-Heinemann, 2017
- [4] P. Corke, Robotics, vision and control, Fundamental algorithms in MATLAB. Berlin Heidelberg: Springer, 2011
- [5] S. Thrun, W. Burgard in D. Fox, Probabilistic robotics. 4, illustr. MIT Press, 2005