موقعیت‌یابی ربات خودمختار بر اساس الگوریتم تکامل تفاضلی

رمضان هاونگی

استادیار، دانشگاه بیرجند، Havangi@Birjand.ac.ir

تاریخ دریافت : ۱۳۹۵/۰۲/۲۰ تاریخ پذیرش :۱۳۹۵/۰۷/۰۳

چکیده

موقعیت‌یابی بر اساس فیلتر ذره‌ای یکی از مهمترین روش‌ها است. با وجود این، این روش دارای مشکل هزینه محاسباتی بالا، عدم سازگاری و تباهیدگی است. در این مقاله برای رهایی از این مشکلات به مسئله موقعیت‌یابی ربات از زاویه جدیدی به مسئله موقعیت یابی نگاه شده است. در روش پیشنهادی، مسئله موقعیت یابی به یک مسئله بهینه‌سازی تبدیل شده و سپس از یک الگوریتم تکامل تفاضلی(DE) برای تخمین موقعیت و جهت ربات استفاده شده است. الگوریتم پیشنهادی در فضای حالت ربات شروع به جستجو می‌کند و موقعیت ربات را با توجه به اطلاعات مسافت پیمایی، فاصله-یاب لیزری و نقشه محیط در هر لحظه بدست می‌آورد . یکی از مزایای موقعیت‌یابی بر اساس روش پیشنهادی در این است که مانند موقعیت‌یابی براساس فیلتر ذره‌ای نیاز به تابع توزیع پیشنهادی و گام نمونه‌برداری مجدد ندارد. بنابراین خطر تباهیدگی کاهش می‌یابد. عملکرد موقعیت‌یابی مبتنی برتکامل تفاضلی با موقعیت‌یابی مبتنی بر فیلتر کالمن توسعه یافته و فیلتر ذره‌ای مورد ارزیابی قرار گرفته است. نتایج نشان می‌دهد که عملکرد موقعیت‌یابی مبتنی بر الگوریتم تکامل تفاضلی بهتر از سایر الگوریتم‌های موقعیت‌یابی می‌باشد.

کلیدواژه

موقعیت‌یابی، فیلتر کالمن توسعه یافته،فیلتر ذره‌ای، تکامل تفاضلی

مقدمه

کاربردهای زیادی وجود دارد که یک ربات متحرک خودمختار به جای انسان ماموریت‌هایی خطرناک را انجام می‌دهد. برای اجرای چنین اهدافی به صورت خوکار، مسئله موقعیت‌یابی ربات یک موضوع کلیدی است. موقعیت‌یابی، تعیین موقعیت[1] و جهت[2] ربات با استفاده از اطلاعات اندازه‌گیری شده توسط سنسورهای نصب شده بر روی آن و نقشه محیط می‌باشد[۳-۱]. دو روش کلی برای موقعیت‌یابی ربات وجود دارد:[۳-۴] موقعیت‌یابی نسبی و مطلق. موقعیت‌یابی نسبی، تعیین موقعیت ربات با استفاده از سنسورهای داخلی مانند انکدرها، شتاب-سنج‌ها و ژیروسکوپ‌ها است. مهم‌ترین عیب این روش افزایش خطای موقعیت‌یابی در طول زمان است. موقعیت‌یابی مطلق، تعیین موقعیت و وضعیت ربات با استفاده از سنسورهای خارجی نصب‌شده بر روی آن مانند فاصله‌یاب‌لیزری و نقشه محیط است. به‌منظور بهره‌مندی از مزایای موقعیت‌یابی نسبی و مطلق ، معمولاً از ترکیب آنها جهت موقعیت‌یابی استفاده می‌شود[۳-۱].

عمومی‌ترین روش برای حل مسئله موقعیت‌یابی فیلتر بیز است [۲-۱]. از نقطه نظر فیلتر بیز، مسئله موقعیت‌یابی ربات تخمین تابع چگالی احتمال پسین موقعیت‌یابی است. با وجود این، در عمل پیاده سازی فیلتر بیز مشکل است. بسته به مدل فرآیند و اندازه‌گیری مسئله موقعیت‌یابی روش‌های مختلفی برای پیاده‌سازی عملی فیلتر بیز وجود دارد [۴]. با توجه به این‌که در مسئله موقعیت‌یابی ربات مدل فرآیند و اندازه‌گیری غیرخطی است دو روش موقعیت‌یابی وجود دارد: موقعیت‌یابی بر اساس فیلتر کالمن توسعه‌یافته (EKF)[3] و موقعیت‌یابی بر اساس فیلتر ذره‌ای (PF)[4].

[1] Position
[2] Orientation

[3] Extended Kalman filter
[4] Particle filter

موقعیت‌یابی بر اساس EKF یکی از شناخته‌ترین و با سابقه-ترین روش‌ها است که تاکنون تحقیقات گسترده‌ای بر روی آن صورت گرفته است[۴-۸]. با وجود این ، با توجه به خطی‌سازی معادلات غیرخطی حرکت ربات، موقعیت‌یابی بر اساس EKF از دقت خوبی برخوردار نخواهد بود و امکان ناسازگاری و ناپایداری وجود دارد[۳-۸]. برای حل این مشکلات موقعیت‌یابی ربات مبتنی بر فیلتر ذره‌ای مطرح شده است [۹-۱۱].

فیلتر ذره‌ای در اصل یک پیاده سازی مبتنی بر روش مونت کارلو است که بطور وسیعی در تخمین سیستم‌های غیرخطی و غیرگوسی کاربرد دارد[۱۲] . در موقعیت‌یابی مبتنی بر فیلتر ذره‌ای چگالی احتمال پسین(PDF)[5]موقعیت‌یابی با مجموعه وزن داده‌شده‌ای از ذرات تخمین زده می‌شود. تا کنون محققان زیادی بر روی موقعیت‌یابی مبتنی بر فیلتر ذره‌ای کار نموده‌اند و بهینه سازی‌هایی بر روی آن انجام داده‌اند[۱۳-۱۸].

تاکنون محققان زیادی بر روی بهبود موقعیت‌یابی بر اساس فیلتر ذره‌ای تلاش نموده‌اند[۱۹-۲۰]. با وجود این، یکی از مهمترین مشکلات الگوریتم‌های موقعیت‌یابی مبتنی بر فیلتر ذره‌ای این است که در طول زمان خطر تباهیدگی و عدم سازگاری وجود دارد[۲۱-۲۲]. این مسئله مربوط به انتخاب تابع توزیع پیشنهادی و روش نمونه‌برداری مجدد است. در حالت کلی طراحی تابع توزیع پیشنهادی بهینه مشکل است. بعلاوه، نمونه برداری مجدد موجب دور ریختن برخی ذرات و کپی کردن برخی ذرات دیگر خواهد شد که سبب از بین رفتن تنوع میان ذرات و تباهیدگی می‌شود[۲۱-۲۲].

برای رهایی از این مشکلات، در این مقاله، به موقعیت‌یابی ربات از زاویه جدیدی توجه شده است. در روش ارائه شده مسئله موقعیت‌یابی ربات به یک مسئله بهینه‌سازی تبدیل شده و سپس از الگوریتم تکامل تفاضلی برای حل این مسئله استفاده شده است. الگوریتم تکامل تفاضلی یک الگوریتم اتفاقی جمعیتی مبتنی بر استراتژی جستجو است که قابلیت مواجه شده با توابع هدف غیر دیفرانسیل پذیر، چند متغیره و غیرخطی را دارد[۲۳-۲۵].در حالی که DE شباهت‌هایی با سایر الگوریتم‌ها تکاملی دارد، این الگوریتم اساسا" از این جهت که از اطلاعات فاصله و جهت جمعیت فعلی برای هدایت فرآیند جستجو استفاده می‌کند با سایر الگوریتم‌ها متفاوت است. از ویژگی‌های این الگوریتم این است که DE به آسانی پیاده‌سازی می شود و تعداد پارامترهای کمی برای تنظیم کردن دارد. بعلاوه، در DE ، ذرات با همدیگر تشریک مساعی دارند[۲۳-۲۵].

ساختار بقیه مقاله به شرح زیر است. در بخش۲ ضمن تعریف مسئله موقعیت‌یابی ربات روش‌های عمده موقعیت‌یابی بطور مختصر بررسی شده است. موقعیت‌یابی بر اساس الگوریتم تکامل تفاضلی در بخش ۳ ارائه شده است. در بخش۴ الگوریتم موقعیت‌یابی پیشنهادی تحت شرایط مختلف مورد ارزیابی قرار گرفته است. در بخش ۵ نتیجه‌گیری ارایه شده است.

موقعیت‌یابی ربات

برای توصیف مسئله موقعیت‌یابی ربات، فرض می‌شود که محیط استاتیک اطراف ربات را بتوان با یک بردار θ که توصیف کننده نشانه‌های نقشه محیط است نشان داد. با این فرض، معادلات گسسته استاندارد توصیف کننده مسئله موقعیت‌یابی به صورت زیر است:

$$\begin{aligned} x_t &= f(x_{t-1}, u_t) + \omega_{t-1} \\ y_t &= h(x_t, \theta) + v_t \end{aligned} \qquad (۱)$$

که u_t بردار کنترل و x_t بردار حالت ربات است که می‌توان آن را بصورت یک فرآیند مارکوف با توزیع اولیه $p(x_t)$ و مدل حرکت $p(x_t \mid x_{t-1}, u_t)$ مشخص کرد. بعلاوه ω_t نویز فرآیند، v_t نویز اندازه‌گیری و y_t بردار اندازه‌گیری‌ها است. توابع f و h توابع غیر خطی هستند. مدل احتمالاتی حرکت ربات و اندازه‌گیری به صورت زیر است[۱-۳]:

$$p(x_t \mid x_{t-1}, u_{t-1}) \qquad (۲)$$

$$p_\theta(y_t \mid x_t) \qquad (۳)$$

با توجه به اینکه در موقعیت‌یابی نقشه محیط (بردار θ) معلوم است، مسئله موقعیت‌یابی از دیدگاه تئوری بیزین عبارت از تخمین تابع چگالی احتمال پسین $p(x_t \mid Y_t)$ است. به عبارت دیگر مسئله موقعیت‌یابی عبارت است از تخمین توزیع پسین حالت‌های سیستم (x_t) به شرط همه اندازه‌گیری‌های قابل دسترس ($Y_t = \{y_0, y_1, ..., y_t\}$) می‌باشد. فیلتر بیز تابع

[5] Probability density function

چگالی احتمال‌پسین را طی دو گام بصورت زیر محاسبه می-کند[۱-۳]:

۱- بروز رسانی[۶]

$$p(x_t \mid Y_t) = \frac{p(y_t \mid x_t)p(x_t \mid Y_{t-1})}{p(y_t \mid Y_{t-1})} \quad (۴)$$

که توابع $p(y_t \mid Y_{t-1})$ و $p(x_t \mid Y_{t-1})$ به صورت زیر هستند:

$$p(y_t \mid Y_{t-1}) = \int p(y_t \mid x_t)p(x_t \mid Y_{t-1})dx_t$$

$$p(x_t \mid Y_{t-1}) = \int p(x_t \mid x_{t-1}, u_{t-1})p(x_{t-1} \mid Y_{t-1})dx_{t-1}$$

توابع $p(y_t \mid x_t)$ و $p(x_t \mid x_{t-1}, u_{t-1})$ به ترتیب با توجه به معادله اندازه‌گیری، معادلات سیستم و نویز اندازه‌گیری v_k و نویز پروسه ω_t قابل محاسبه است.

۲- پیش بینی[۷]

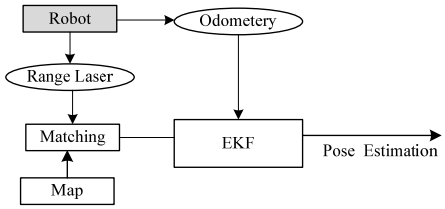
تابع چگالی احتمال پسین به صورت زیر بروز رسانی می‌شود:

$$p(x_{t+1} \mid Y_t) = \int p(x_{t+1} \mid x_t, Y_t)p(x_t \mid Y_t)dx_t = \int p(x_{t+1} \mid x_t)p(x_t \mid Y_t)dx_t \quad (۵)$$

ملاحظه می‌شود که فیلتر بیز راه‌حل تحلیلی و بسته برای موقعیت یابی ارائه نمی‌کند. در نتیجه مناسب پیاده‌سازی نمی‌باشد. با توجه به اینکه معادلات حرکت و اندازه‌گیری ربات غیرخطی می‌باشند، روش‌های زیر بهینه برای پیاده سازی فیلتر بیز استفاده می‌شوند. مهمترین روش پارامتریک و غیر پارامتریک برای پیاده‌سازی فیلتر بیز به ترتیب فیلتر کالمن توسعه یافته و فیلتر ذره‌ای است[۱].

موقعیت‌یابی بر اساس فیلتر کالمن توسعه یافته

موقعیت‌یابی بر اساس EKF یکی از قدیمی‌ترین و ساده-ترین روش‌ها است که تاکنون کارهای تحقیقاتی زیادی بر روی آن انجام شده است[۴-۸].در موقعیت‌یابی مبتنی بر EKF، تابع پسین با یک توزیع گوسی چند متغیره قابل بیان است[۱]. شکل (۱) فلوچارت موقعیت‌یابی با EKF را نشان می‌دهد.



شکل ۱. فلوچارت موقعیت یابی با EKF

بطور خلاصه الگوریتم موقعیت‌یابی با EKF دارای گام‌های زیر است:

۱- پیش‌بینی

در این گام میانگین و کواریانس حالت‌های ربات به صورت زیر پیش‌بینی می‌شود:

$$\hat{x}_t^- = f(x_{t-1}, u_t) \quad (۶)$$

$$P_t^- = \nabla f_t P_{t-1}^- \nabla f_t^T + G_u Q_{t-1} G_u^T \quad (۷)$$

که در آن ∇f_t و G_u به فرم زیر است :

$$\nabla f = \frac{\partial f}{\partial x} \qquad G_u = \frac{\partial f}{\partial u}$$

Q_{t-1} ماتریس کواریانس نویز فرآیند و P_{t-1}^- ماتریس کواریانس خطا است.

۳- تطابق[۸]

در این گام، اندازه‌گیری‌ها با نشانه‌های موجود در نقشه تطابق داده می‌شوند. در صورتی یک اندازه‌گیری با نشانه ای در نقشه تطابق داده می‌شود که نامساوی زیر برقرار باشد:

$$Inn_t S_t Inn_t^T \le G \quad (۸)$$

در این رابطه Inn_t عبارت از اختلاف اندازه‌گیری واقعی(y_t) و اندازه‌گیری تخمین زده ($\hat{y}_t$) شده می‌باشد که به باقیمانده[۹] معروف است.

$$Inn_t = y_t - \hat{y}_t \quad (۹)$$

[6] Update
[7] Prediction
[8] Matching
[9] Residual

S_t کواریانس باقیمانده و به صورت زیر است:

$$S_t = \nabla h_t P_t^- \nabla h_t^T + R_t \ , \ \nabla h = \frac{\partial h}{\partial x} \qquad (۱۰)$$

۵-بروزرسانی

در این مرحله میانگین و کواریانس موقعیت و وضعیت ربات مطابق روابط زیر بروز رسانی می‌شوند:

$$\hat{x}_t = \hat{x}_t^- + K_t(y_t - h(\hat{x}_t^-)) \qquad (۱۱)$$

$$P_t = (I - K_t H_t)P_t^- \qquad (۱۲)$$

که K_t گین EKF و به صورت زیر است:

$$K_t = P_t^- H_t^T (H_t P_t^- H_t^T + R_t)^{-1} \qquad (۱۳)$$

از آنجایی که موقعیت‌یابی با EKF از روابط خطی‌شده مدل غیرخطی حرکت و اندازه‌گیری استفاده می‌کند این خطی سازی می‌تواند سبب ناسازگاری و ناپایداری این روش شود[۱-۴].

موقعیت‌یابی ربات مبتنی بر فیلتر ذره‌ای

این روش بر خلاف الگوریتم موقعیت‌یابی بر اساس EKF هیچ قیدی برای نویزها، اعم از گوسی‌بودن یا غیرگوسی بودن، قائل نیست. موقعیت‌یابی مبتنی بر فیلتر ذره‌ای، توزیع احتمال پسین موقعیت‌یابی $p(x_t \mid Y_t)$ را با مجموعه‌ای از ذرات به همراه وزن‌شان ارائه می‌کند[۱۲]:

$$p(x_t \mid Y_t) \approx \sum_{i=1}^{N} w_t^i \delta(x_t - x_t^i) \qquad (۱۴)$$

که $\delta(.)$ تابع دلتای دایراک و $w_t^i > 0$ وزن مربوط به x_t^i است که $\sum_{i=1}^{N} w_t^i = 1$ است. با توجه به این‌که نمونه‌برداری مستقیم از تابع توزیع هدف $p(x_t \mid Y_t)$ ممکن نیست، از روش نمونه‌برداری پراهمیت[1] استفاده می‌شود[۱۲].

در این روش نمونه‌برداری پر اهمیت به جای نمونه‌برداری از تابع هدف از یک تابع توزیع پیشنهادی نمونه‌برداری می‌شود[۱۲]. در صورتی که تابع توزیع پیشنهادی را به صورت $q(x_t \mid Y_t)$ در نظر گرفته شود، وزن مربوط به ذرات به صورت زیر است[۱۲]:

$$w_t^i = \frac{p(x_t \mid Y_t)}{q(x_t \mid Y_t)} \qquad (۱۵)$$

در حالت بهینه تابع توزیع پیشنهادی $q(x_t \mid Y_t)$ باید خود تابع چگالی احتمال پسین $p(x_t \mid Y_t)$ باشد در غیر این‌صورت واریانس وزن‌های پراهمیت با گذشت زمان افزایش می‌یابد[۱۲]. افزایش واریانس اثر بدی روی دقت تخمین ها دارد و سبب ایجاد پدیده‌ای بنام تباهیدگی[1] می‌شود[۱۲،۱] در صورت تباهیده شدن، بعد از چند تکرار وزن نرمال شده همه ذرات بجز یکی ناچیز می‌شوند. در نتیجه برای بروز رسانی ذراتی تلاش می‌شود که سهم‌شان در تقریب تابع چگالی احتمال پسین $p(x_t \mid Y_t)$، تقریبا" ناچیز است[۱۲-۱۳]. یک اندازه مناسب از تباهیدگی ، اندازه نمونه موثر[2] N_{eff} است که به صورت زیر تعریف می‌شود[۱۳-۱۴]:

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N} (w_t^i)^2} \qquad (۱۶)$$

که w_t^i وزن نرمال شده و به صورت زیر است:

$$w_t^i = \frac{w_t^i}{\sum_{i=1}^{N} (w_t^i)^2}$$

به‌منظور کاهش پدیده تباهیدگی، از نمونه‌برداری‌مجدد استفاده می‌شود[۱۵]. هر جا که پدیده تباهیدگی مشاهده شد نیاز به گام نمونه‌برداری مجدد است. گام نمونه‌برداری مجدد سبب می‌شود که ذرات با وزن کوچک حذف و ذرات با وزن بزرگتر تکثیر شوند[۱۲]. به طور خلاصه الگوریتم موقعیت‌یابی با فیلتر ذره‌ای به صورت زیر است:

۱- نمونه برداری

[1] Importance Sampling

[1] Degeneracy
[2] Effective sample size

ذرات با نمونه‌برداری از توزیع پیشنهادی $q(x_t \mid Y_t)$ تولید می‌شوند.

۲- محاسبه وزن نمونه‌ها

$$w_t^i = \frac{p(x_t \mid Y_t)}{q(x_t \mid Y_t)}$$

۳- نرمال‌سازی وزن‌ها

$$w_t^i = \frac{w_t^i}{\sum_{i=1}^{N} w_t^i}$$

۳- نمونه‌گیری مجدد

ذرات با وزن کمتر با ذرات با وزن بالاتر جایگزین می‌شوند.

یک مشکل موقعیت‌یابی با فیلتر ذره‌ای هزینه محاسباتی بالای آن است. در این روش برای دقت خوب ، تعداد ذرات باید بالا باشد که در نتیجه سبب بالا رفتن بار محاسبات می‌شود. تعداد ذرات کم هم سبب تخمین‌های ضعیف و یا حتی سبب واگرایی می‌شود.

علاوه بر این، یکی از مشکلات موقعیت‌یابی مبتنی بر فیلتر ذره‌ای این است که در طول زمان تباهیده می‌شود. این مسئله مربوط به تابع توزیع پیشنهادی انتخابی و روش نمونه‌برداری مجدد است.

مسئله فقر نمونه‌ها در فرآیند نمونه‌برداری بدلیل عدم تطبیق بین تابع توزیع هدف و تابع توزیع پیشنهادی اتفاق می‌افتد. در حالت کلی طراحی تابع توزیع پیشنهادی بهینه مشکل است. تابع توزیع پیشنهادی زیر بهینه مختلفی در مقالات ارائه شده است[۲۶،۱۲]. برای مثال تابع انتقال و دیگر توابع توزیع زیر بهینه‌ای که وجود دارد[۲۶] . با وجود این ، یک تقریب بهینه برای تابع چگالی احتمال پیشنهادی با استفاده از تکنیک‌های خطی‌سازی محلی غیرممکن است[۱۲]. در حوزه فیلترینگ، نسخه‌های مختلفی از فیلتر ذره‌ای برای حل بهبود کاستی‌های این فیلتر معرفی‌شده‌اند که می‌توان به فیلتر ذره‌ای کمکی، فیلتر ذره‌ای خطی محلی اشاره نمود[۲۷-۲۸].

فرآیند نمونه‌برداری مجدد اگر چه سبب کاهش مسئله تباهیدگی سبب می‌شود اما از طرف دیگر سبب می‌شود که تنوع میان ذرات از بین برود و سبب ایجاد پدیده فقر نمونه شود[۷-۱۳]. این پدیده وقتی نویز پروسه پایین باشد شدید است و سبب می‌شود که همه ذرات به یک نقطه ریزش کنند. در نتیجه ذرات نمی‌توانند به درستی تابع چگالی احتمال پسین را تقریب بزنند و دقت تخمین‌ها کاهش می‌آید.

در این مقاله ، برای دوری از این مشکلات موقعیت‌یابی بر اساس فیلتر ذره‌ای، به مسئله موقعیت‌یابی به صورت یک مسئله بهینه‌سازی نگریسته شده است.

موقعیت‌یابی بر اساس تکامل تفاضلی

در این روش به مسئله موقعیت از دیدگاه ماکزیمم کردن تابع احتمال پسین نگاه شده است. از این نقطه نظر، مسئله موقعیت‌یابی به صورت زیر است[۱۵-۱۸] :

$$\hat{x}_t^{MAP} = \arg\max_x p(x_t \mid Y_t) \qquad (۱۷)$$

از طرفی $p(x_t \mid Y_t)$ را می‌توان به صورت زیر نوشت[۲۷-۲۸]:

$$p(x_t \mid Y_t) = p(y_t \mid x_t)\, p(x_t \mid x_{t-1}, u_{t-1})\, p(x_{t-1} \mid Y_{t-1})$$

با جایگذاری رابطه بالا در (۱۷)، مسئله موقعیت‌یابی را می‌توان به‌صورت زیر بازنویسی کرد:

$$\hat{x}_t^{MAP} = \arg\max_x p(x_t \mid Y_t) = \arg\max_x p(y_t \mid x_t)\, p(x_t \mid x_{t-1}, u_{t-1})\, p(x_{t-1} \mid Y_{t-1}) \qquad (۱۸)$$

با در نظر گرفتن فرض ماکوف بودن و iid بودن متغیرها با استفاده از قضیه حد مرکزی و ساده‌سازی داریم[۲۸]:

$$\hat{x}_t^{MAP} = \arg\max_x \prod_{i=1}^{t} p(y_i \mid x_i) \prod_{i=1}^{t} p(x_i \mid x_{i-1}, u_{t-1})\, p(x_{t-1} \mid Y_{t-1}) \qquad (۱۹)$$

بنابراین از دیدگاه ماکزیمم کردن تابع احتمال پسین به مسئله موقعیت‌یابی مانند یک مسئله بهینه سازی می‌توان نگاه کرد. در صورتی که $f(x)$ بیان کننده تابع هدف باشد داریم:

$$f(x)=\prod_{i=1}^{t} p(y_i \mid x_i) \prod_{i=1}^{t} p(x_i \mid x_{i-1},u_{t-1}) p(x_{t-1} \mid Y_{t-1}) \quad (۲۰)$$

با توجه به معادلات سیستم و اندازه‌گیری معادله $f(x)$ را می‌توان به صورت زیر نوشت:

$$f(x)=\prod_{i=1}^{t} p_v(y_i \mid x_i) \prod_{i=1}^{t} p_w(x_i \mid x_{i-1},u_{t-1}) p(x_{t-1} \mid Y_{t-1}) \quad (۲۱)$$

در رابطه بالا p_v و p_w به‌ترتیب بیان‌کننده تابع چگالی احتمال نویز اندازه‌گیری و فرآیند است. در این صورت مسئله موقعیت‌یابی ربات عبارت است از پیدا کردن x در فضای شدنی مسئله موقعیت‌یابی به گونه‌ای که تابع $f(x)$ در میان همه x ها ماکزیمم شود. به عبارتی موقعیت ربات x باید به گونه‌ای باشد که ضمن ماکزیمم کردن تابع هدف $f(x)$ در معادلات (۱) صدق نماید. با لگاریتم گرفتن از طرفین معادله (۳۲) داریم:

$$f_0(t)=\log f(x)=\log p_v(y_t \mid x_t) + \log p_w(x_t \mid x_{t-1},u_{t-1}) + \sum_{i=1}^{t-1} \log p_v(y_i \mid x_i) + \sum_{i=1}^{t-1} \log p_v(x_i \mid x_{i-1},u_{t-1}) + \log p(x_{t-1} \mid Y_{t-1}) \quad (۲۲)$$

از طرفی

$$f_0(t-1)=\sum_{i=1}^{t-1} \log p_v(y_i \mid x_i) + \sum_{i=1}^{t-1} \log p_w(x_i \mid x_{i-1},u_{t-1}) + \log p(x_{t-1} \mid Y_{t-1}) \quad (۲۳)$$

بنابراین داریم:

$$f_0(x_t)=\log p_v(y_t \mid x_t) + \log p_w(x_t \mid x_{t-1},u_{t-1}) + f_0(t-1) \quad (۲۴)$$

بنابراین مسئله ماکزیمم‌سازی مورد نظر یک مسئله بهینه‌سازی بازگشتی است.

اساس فرآیند بهینه‌سازی بازگشتی در (۲۴) بر مبنای اصل بلمن است و مسیر بهینه ربات با استفاده از برنامه ریزی پویای بلمن محاسبه می‌شود. مطابق اصل بهینه‌سازی بلمن، یک سیاست بهینه دارای این خاصیت است که شرایط اولیه و تصمیم اولیه هرچه باشد تصمیم‌های باقیمانده باید یک سیاست بهینه نسبت به حالت منتجه از تصمیم اولیه داشته باشد. بنابرین مسئله بهینه سازی مورد نظر به صورت زیر خواهد شد:

$$\hat{f}(x_t)=\log p_v(y_t \mid x_t)+\log p_w(x_t \mid x_{t-1},u_{t-1})$$

در معادله بالا ترم $p_v(y_t \mid x_t)$ ، تابع درستنمایی[۳]است و به صورت زیر است:

$$p_v(y_t \mid x_t)=\frac{1}{(2\pi)^{m_y/2}|R|^{\frac{1}{2}}} \exp\left(-\frac{(y_t-\hat{y}_t)R^{-1}(y_t-\hat{y}_t)^T}{2}\right)$$

که R کواریانس نویز اندازه‌گیری است:

$$E[v_t v_t^T]=R$$

ترم $p_w(x_i \mid x_{i-1},u_{t-1})$ با توجه به مدل فرآیند و فرم زیر است:

$$p_w(x_t \mid x_{t-1},u_{t-1})=\frac{1}{(2\pi)^{n_x/2}|Q|^{\frac{1}{2}}} \exp\left(-\frac{(x_t-\hat{x}_t)Q^{-1}(x_t-\hat{x}_t)^T}{2}\right)$$

که در این معادله $\hat{x}_i$ به صورت

$$\hat{x}_t=f(\hat{x}_{t-1},u_t)$$

و Q کواریانس نویز فرآیند است:

$$E[w_t w_t^T]=Q$$

. با جایگذاری $p_v(z_t \mid \hat{x}_t)$ و $p_w(x_i \mid x_{i-1},u_{t-1})$ در معادله (۲۴) داریم:

[1] Liklihood [3]

$$\hat{f}(x_t)=\log((2\pi)^{m_y/2}|R|^{\frac{1}{2}})-\frac{(y_t-\hat{y}_t)R^{-1}(y_t-\hat{y}_t)^T)}{2}$$
$$+\log((2\pi)^{n_x/2}|Q|^{\frac{1}{2}})-\frac{(x_t-\hat{x}_t)Q^{-1}(x_t-\hat{x}_t)^T}{2}$$

با توجه به اینکه مقادیر ثابت در مینیمم سازی تاثیری ندارند تابع هدف به صورت زیر کاهش پیدا می‌کند:

$$\tilde{f}_0(x_t)=\frac{1}{2}(y_t-\hat{y}_t)R^{-1}(y_t-\hat{y}_t)^T$$
$$+\frac{1}{2}(x_t-\hat{x}_i)Q^{-1}(x_t-\hat{x}_t)^T \quad (۲۵)$$

که Q کواریانس نویز فرآیند، R کواریانس نویز اندازه‌گیری، y_t اندازه‌گیری و $\hat{y}_t$ اندازه‌گیری تخمین‌زده شده است. بنابراین بطور خلاصه مسئله موقعیت‌یابی ربات به مسئله بهینه سازی زیر تبدیل می‌شود:

$$\arg\min_{x_t}\tilde{f}_0(x_t) \quad (۲۶)$$

ملاحظه می‌شود، این معادله غیرخطی است و بنابراین امکان وجود مینیمم محلی در آن وجود دارد. با بکار بردن روش‌های سنتی، برای حل این مسئله بهینه‌سازی معمولا" یک جواب محلی بدست می‌آید. این ضعف بیان می‌کند که این روش برای این مسئله مناسب نمی‌باشد. برای دوری از این مشکل در اینجا از الگوریتم تکامل تفاضلی استفاده شده است . این بدان دلیل است که حجم پردازش این الگوریتم کمتر از سایر الگوریتم‌ها است. از طرف دیگر در این الگوریتم ذرات تجربه جمعی خود را در اختیار خود قرار می‌دهند و در نتیجه در این الگوریتم ذرات با سرعت بیشتری به نقطه بهینه هدایت می‌شوند.

الگوریتم تکامل تفاضلی

در سالهای اخیر الگوریتم تکامل تفاضلی به عنوان روشی قدرتمند و سریع برای مسائل بهینه‌سازی در فضاهای پیوسته معرفی شده است[۴] و توانایی خوبی در بهینه‌سازی توابع غیرخطی مشتق‌ناپذیر دارد. همانند دیگر الگوریتم‌های تکاملی، این الگوریتم با ایجاد یک جمعیت اولیه شروع به کار می‌کند. سپس با اعمال عملگرهایی شامل ترکیب، جهش و تقاطع، نسل نوزاد تشکیل شده و در مرحله بعد که مرحله انتخاب نام دارد، نسل نوزاد با نسل والد از برای میزان شایستگی که توسط تابع هدف سنجیده می‌شود، مقایسه می‌گردد. سپس بهترین اعضا به عنوان نسل بعدی وارد مرحله بعد می‌گردند. این عمل تا رسیدن به نتایج مطلوب ادامه می‌یابد.

الگوریتم تکامل تفاضلی شبیه به سایر الگوریتم‌های تکاملی از اپراتورهای تکاملی برش و جهش استفاده می‌کند. این الگوریتم از اطلاعات فاصله و جهت جمعیت فعلی برای هدایت فرآیند جستجو استفاده می‌کند. اپراتور برش مهمترین اپراتور در الگوریتم تکامل تفاضلی است. این اپراتور در ابتدا برای ایجاد بردار هدف و سپس برای تولید فرزند استفاده می‌شود. به طور خلاصه این الگوریتم دارای مراحل زیر است[۴]:

۱- ایجاد جمعیت اولیه

جمعیت اولیه $C(0)$ بصورت تصادفی ایجاد می‌شود. تعداد ژن‌های جمعیت در هر کروموزم بستگی به تعداد متغیرهای تابع معیار بستگی دارد.

۲-محاسبه تابع معیار

برای هر کروموزم $(x_i(t))$ از جمعیت $(C(t))$ تابع معیار حساب می‌شود.

۳- تشکیل بردار هدف

بردار هدف $u_i(t)$ برای هر کروموزم با اعمال اپراتور جهش به صورت زیر تشکیل می‌شود:

$$u_i(t)=x_i(t)+\beta(x_{i_2}(t)-x_{i_3}(t)) \quad (۲۷)$$

i_2، i_3 به صورت تصادفی به گونه‌ای انتخاب می‌شوند که

$$i\neq i_1\neq i_2\neq i_3 \quad (۲۸)$$

i نشان دهنده کروموزم i از جمعیت $C(t)$ است.

۴-برش

بین بردار جهش یافته $u_i(t)$و عضو هدف که در مرحله اول انتخاب شد، یک برش صورت می گیرد و ژن های فرزند $x_i'(t)$ به صورت زیر تعیین می‌شوند:

$$x'_{ij}(t)=\begin{cases}u_{ij}(t) & j\in J\\ x_{ij}(t) & j\notin J\end{cases} \qquad (۲۹)$$

که J نشان دهنده ژن‌های کروموزم iـام است که تحت اپراتور برش قرار می‌گیرند.

۵- انتخاب

در صورتی که تابع معیار فرزند $x'_i(t)$ از $x_i(t)$ بهتر باشد $x'_i(t)$ به نسل بعد منتقل خواهد شد. در غیر این صورت خود $x_i(t)$ به نسل بعد منتقل خواهد شد.

الگوریتم موقعیت‌یابی مبتنی بر تکامل تفاضلی

همانطور که گفته شد مسئله موقعیت‌یابی ربات با مسئله بهینه‌سازی غیرخطی رابطه (۲۶) معادل است. برای حل این مسئله با استفاده از الگوریتم تکامل تفاضلی، ابتدا جمعیت اولیه‌ای متشکل از کروموزوم‌ها تشکل می‌شود که تعداد ژن‌های آن به تعداد حالتهای سیستم (یعنی موقعیت و جهت) است که باید تخمین زده شوند.

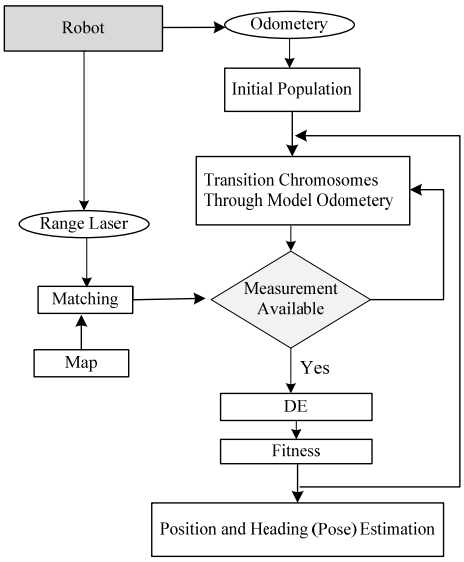
فرض کنید، الگوریتم DE از بردار $x^k_{t,i}$ برای ارائه هر کاندید جواب i در تکرار k و گام زمانی داده شده t استفاده می‌کند. برای هر بردار هدف $x^k_{t,i}$ ، الگوریتم DE یک بردار آزمایش u را به صورت زیر تولید می‌کند:

$$u_t = x^k_{t,i} + \beta(x^k_{t,Global} - x^k_{t,i_1}) \qquad (۳۰)$$

که $x^k_{t,Global}$ کرومزوم با بهترین تابع معیار، $x^k_{t,i}$ بردار هدف برای بردار هدف برای آشفتن در تکرار k است. x^k_{t,i_1} عضوی از جمعیت است که به صورت تصادفی طوری انتخاب شده است که i و i_1 با هم مساوی نباشند. پارامتر β ضریب مقیاس است که دامنه واریانس تفاضل $(x^k_{t,i_2}-x^k_{t,i_3})$ را کنترل می‌کند. برای افزایش تنوع بردارهای نسل جدید، یک مکانیزم برش به صورت زیر استفاده شده است:

$$x'_{t,ij}=\begin{cases}u_{t,ij} & \text{if } j\in J\\ x_{t,ij} & \text{otherwise}\end{cases}$$
$$x'_{t,i}=\{x'_{t,ij}\},\ u_{t,ij}=\{u_{t,i}\} \qquad (۳۱)$$
$$x_{t,i}=\{x_{t,ij}\},\ j=1\ldots D$$

که $x_{t,ij}$ اشاره به j امین المان بردار $x_{t,i}$ دارد و J مجموعه نقاط برش است. برای برش دوجمله‌ای، نقاط برش بطور تصادفی از مجموعه نقاط برش ممکن $\{1,2,\ldots,D\}$ انتخاب می‌شود که D ابعاد مسئله است.



شکل۳. فلوچارت روش پیشنهادی برای موقعیت‌یابی

برای تصمیم گرفتن درباره اینکه x'_t باید عضوی از نسل $t+1$ باشد، بردار جدید با x_t مقایسه می‌شود. اگر بردار x'_t مقداری بهتر برای تابع معیار بدست آورد، در نسل جدید x_t جایگزین x'_t خواهد شد. در غیر این‌صورت مقدار قدیمی x_t برای نسل جدید حفظ می‌شود. تابع هدف به صورت (۲۵) محاسبه می‌شود. وقتی جمعیت جدید تولید می‌شود، فرآیند برش، جهش و انتخاب تا جایی تکرار می‌شود که مینیمم پیدا شود یا به مینیمم از قبل تعیین شده برسد. وقتی بهترین مقدار برازندگی به یک آستانه مشخص رسید، تکرارها متوقف می‌شوند. بطور خلاصه فلوچارت الگوریتم موقعیت یابی ربات بر اساس تکامل تفاضلی به ربات بصورت شکل(۳) است.

شبیه سازی و نتایج

حرکت یک ربات دوچرخه در نظر می‌گیریم. ربات فرامین کنترلی را اجرا و اندازه‌گیری‌ها را از نشانه‌هایی که در طبیعت هستند جمع آوری می‌کند. معادلات گسسته موقعیت‌یابی ربات به صورت زیر است:[۳-۱]:

$$\begin{bmatrix} x_t \\ y_t \\ \phi_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \phi_{t-1} \end{bmatrix} + \begin{bmatrix} (V+v_v)\cos(\phi_{t-1}+[\gamma+v_\gamma]) \\ (V+v_v)\sin(\phi_{t-1}+[\gamma+v_\gamma]) \\ \frac{(V+v_v)}{B}\sin(\gamma+v_\gamma) \end{bmatrix} dt$$

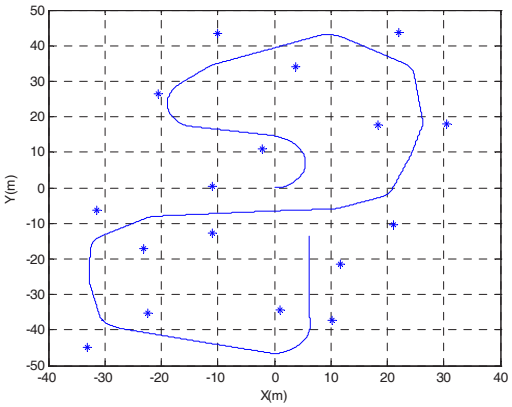
در این معادله، (x,y) موقعیت ربات، ϕ سمت ربات، B طول محور ربات (که چرخ جلو را به چرخ عقب متصل می‌کند) است. ورودی کنترل u شامل سرعت خطی V و جهت چرخ فرمان γ است:

$$u = \begin{bmatrix} V \\ \gamma \end{bmatrix}$$

همچنین فرض شده است که ربات مجز به فاصله یاب‌لیزی است که در جلوی ربات نصب شده است. فاصله‌یاب لیزی فاصله r_i و زاویه سمت θ_i از نشانه مشاهده شده را اندازه‌گیری می‌کند معادلات اندازه‌گیری از نشانه i ام به صورت زیر است:

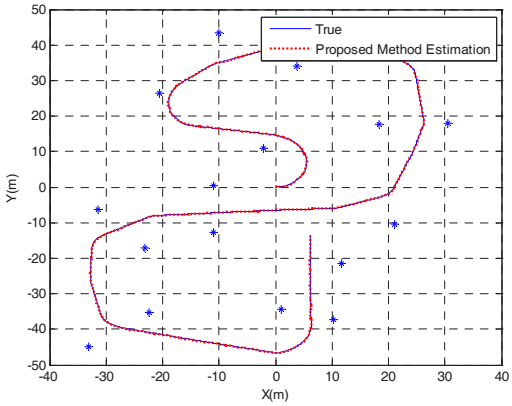
$$z_t = h(.) = \begin{bmatrix} r_i \\ \theta_i \end{bmatrix} = \begin{bmatrix} \sqrt{(x-x_i)^2+(y-y_i)^2} + \omega_r \\ \tan^{-1}\frac{y-y_i}{x-x_i} - \phi + \omega_\theta \end{bmatrix}$$

که (x_i, y_i) موقعیت نشانه مشاهده در نقشه است.

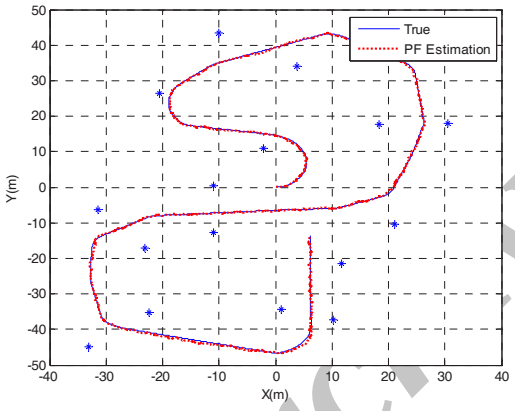


شکل۴. مسیر نامی ربات و نقشه محیط آن

شکل(۴) مسیر حرکت ربات و نقشه محیط را نشان می‌دهد. در این شکل مسیر واقعی ربات و نقشه محیط نشان داده شده است. نشانه‌های موجود در نقشه با علامت * مشخص شده است. سرعت ربات ۳m/s و ماکزییم زاویه فرمان[۱۴] آن درجه ۳۰ در نظرگرفته می‌شود. ربات از مبدا و در خلاف جهت عقربه‌های ساعت شروع به حرکت می‌کند و دوباره به سمت سمت نقطه شروع باز می‌گردد.



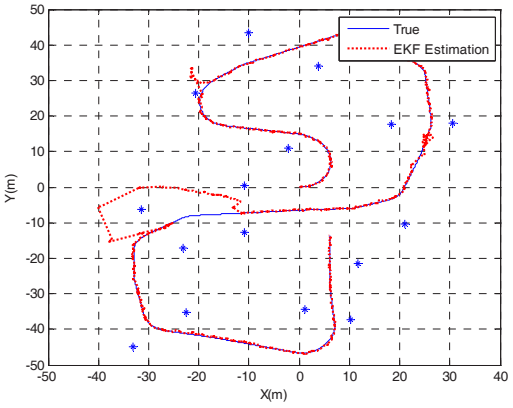
شکل۵. مسیر واقعی و تخمین زده شده توسط روش پیشنهادی



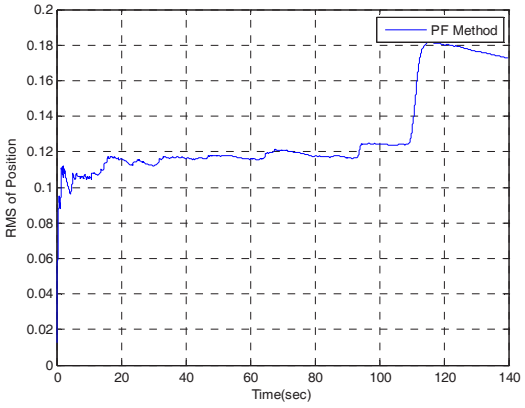
شکل۶. مسیر واقعی و تخمین زده شده توسط PF

شکل‌های(۵) تا (۷) نتایج بدست آمده از موقعیت‌یابی بر اساس روش پیشنهادی ، فیلتر ذره‌ای و فیلتر کالمن توسعه یافته را نشان می‌دهند. بطور واضح دیده می‌شود که نتایج عملکرد روش پیشنهادی بهتر از سایر روش ها است. به عبارت دیگر، در موقعیت‌یابی با الگوریتم پیشنهادی، مسیر تخمین زده شده وسیله تا حد امکان نزدیک به مسیر واقعی می‌باشد.

[1] Steering angle

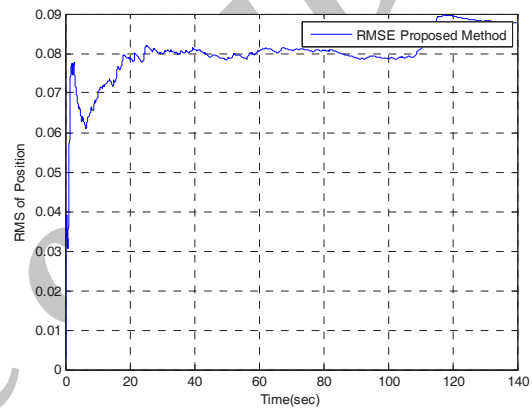


شکل ۷ . مسیر واقعی و تخمین زده شده توسط EKF

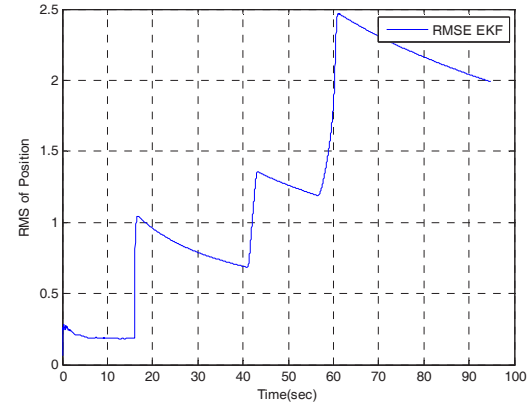


شکل۱۰. موقعیت‌یابی براساس PF

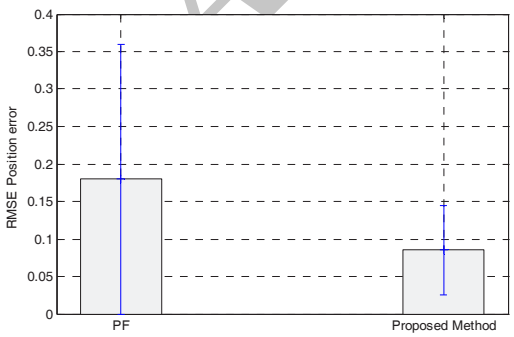
جذر مربع خطا ی(RMSE) تخمین موقعیت الگوریتم‌های مختلف نسبت به زمان درشکل‌های(۸-۱۰) و میانگین و کواریانس آن در شکل(۱۱) نشان داده شده است. RMSE موقعیت از روش مونت کارلو با ۵۰ بار اجرا بدست آمده است. همچنین تعداد ذرات در الگوریتم اجتماع ذرات و فیلتر ذره‌ای N=30 انتخاب شده باشد. همانطورکه از شکل‌ها مشاهده می-شود عملکرد روش موقعیت‌یابی پیشنهادی بهتر از دو الگوریتم دیگر است. به دلیل غیرخطی بودن بالای معادله سیستم و اندازه‌گیری، بدترین تخمین مربوط به موقعیت‌یابی بر اساس فیلترکالمن توسعه‌یافته است. غیرخطی بودن سیستم سبب می‌شود که تقریب خطی سیستم بد شود و در نتیجه تابع توزیع احتمال پسین از حالت گوسی بودن بسیار منحرف شود و در نتیجه دقت تخمین‌های بدست آمده از فیلترکالمن توسعه یافته کاهش یابد. یکی از مهمترین خصوصیات فیلتر پیشنهادی این است که در تعداد تکرار کم به خوبی عمل می‌کند.



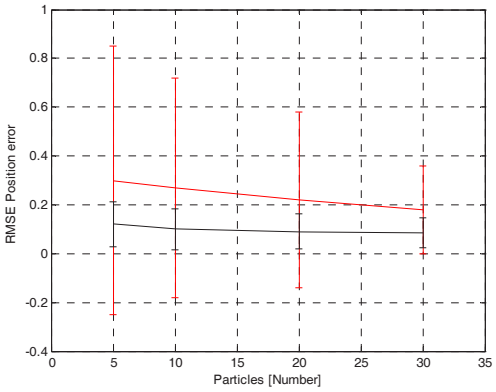
شکل۸. موقعیت‌یابی براساس روش پیشنهادی



شکل۹. موقعیت‌یابی براساس EKF



شکل ۱۱.RMSE موقعیت ربات



شکل۱۲. مقایسه موقعیت یابی با PF و روش پیشنهادی

با تعداد مختلف ذرات

سپس عملکرد موقعیت یابی با PF و روش پیشنهادی با تعداد ذرات مختلف با هم مقایسه شده است. در شکل (۱۲) مشخص است که RMSE موقعیت در موقعیت‌یابی با روش پیشنهادی وابستگی کمتری به تعداد ذرات دارد در حالی که RMSE خطای موقعیت در موقعیت یابی با PF وابسته به تعداد ذرات است.

جدول۱. هزینه محاسباتی الگوریتم‌ها

Number of Particles	Processing time		RMSE	
	PF	Proposed Method	PF	Proposed Method
30	35	52	0.18	0.085
20	32	44	0.22	0.09
10	26	32	0.27	0. 1
5	23	26	0.3	0.12

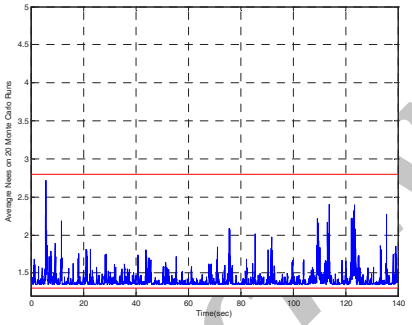
هزینه محاسباتی الگوریتم‌ها با انجام شبیه سازی‌ها روی محیط Matlab بررسی شده است. از جدول ۱ مشاهده می‌شود که حجم محاسبات موقعیت‌یابی با الگوریتم پیشنهادی تقریبا" نزدیک به موقعیت‌یابی با PF است. همچنین، روش پیشنهادی برای بدست آوردن دقت برابر با موقعیت‌یابی با PF به تعداد ذرات کمتری نیاز دارد.

برای بررسی پایداری موقعیت‌یابی با روش پیشنهادی،سازگاری آن بررسی می‌شود. مسئله سازگاری یکی از موضوعات مهم در روش‌های موقعیت‌یابی است. معیارهای مختلفی برای ارزیابی سازگاری فیلتر ذره‌ای وجود دارد [۳۱] . در این مقاله از معیار تخمین نرمالیزه شده مربع خطا(NEES)[۵]برای ارزیابی سازگاری استفاده شده است. تخمین نرمالیزه شده مربع خطا بصورت زیر تعریف می‌شود[۳۱]:

$$\varepsilon_t = (x_t - \hat{x}_t)^T \hat{P}_t^{-1} (x_t - \hat{x}_t)$$

که x_t مقدار واقعی حالت ربات، $\hat{x}_t$ مقدار تخمین زده شده حالت ربات و $\hat{P}_t$ کواریانس خطای تخمین است. سازگاری فیلتر با چندین بار اجرای روش و محاسبه متوسط NEES بررسی می‌شود. برای M بار اجرای مونت کارلو متوسط NEES بصورت زیر محاسبه می‌شود:

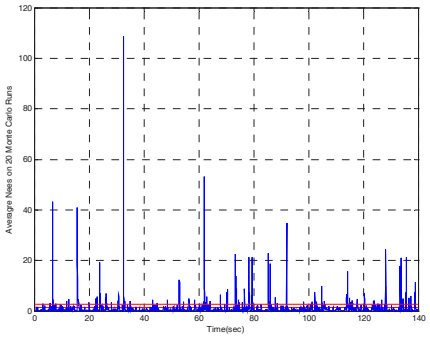
$$\bar{\varepsilon}_t = \frac{1}{M} \sum_{i=1}^{M} \varepsilon_{i_t}$$



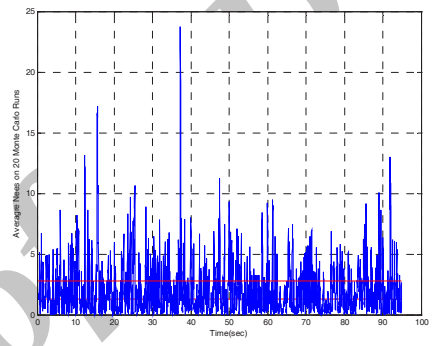
شکل۱۳: ساز گاری موقعیت‌یابی بر اساس روش پیشنهادی

برای موقعیت دوبعدی وسیله، با ۲۰ شبیه سازی مونت کارلو، دو سمت احتمال ۹۵ درصد برای $\bar{\varepsilon}_t$ به بازه [۱,۳,۲,۷۹]محدود می‌شود. شکل‌های (۱۳) تا (۱۵) سازگاری روش‌های موقعیت‌یابی را نشان می‌دهند. نتایج نشان می‌دهند که در موقعیت‌یابی با روش پیشنهادی $\bar{\varepsilon}_t$ محدود به بازه [۱,۳,۲,۷۹] است و در نتیجه سازگار باقی می‌ماند، در حالیکه NEES در موقعیت‌یابی با PF و EKF ناسازگار است. این بدان دلیل است که عملکرد روش پیشنهادی به واقعیت نزدیک‌تر است.

[1] Normalized estimation error squared

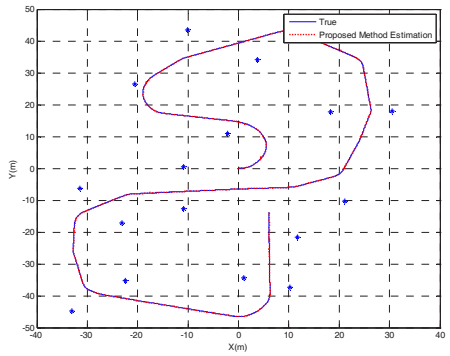


شکل۱۴: ساز گاری موقعیت‌یابی براساس PF

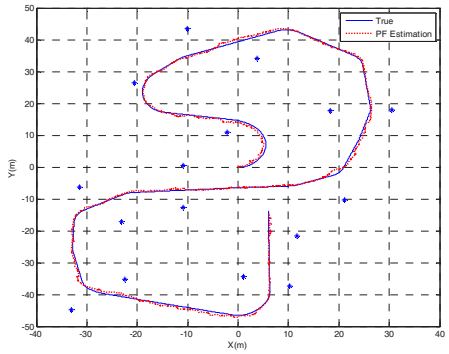


شکل۱۵: ساز گاری موقعیت‌یابی براساس EKF

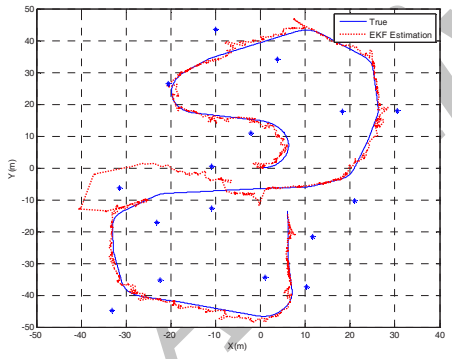
در پایان، عملکرد موقعیت‌یابی بر اساس روش پیشنهادی تحت شرایط نویز غیرگوسی مورد ارزیابی قرار می‌گیرد. برای این منظور، نویز اندازه‌گیری را نویز غیرگوسی فرض و از یک توزیع گاما استخراج شده است.در حالی که نویز فرآیند گوسی سفید فرض می‌شود. نتایج در شکل‌های (۱۶) تا (۲۱) نشان داده شده است. آن بطور واضح دیده می‌شود که عملکرد موقعیت‌یابی بر اساس روش پیشنهادی بهتر از سایر روش‌ها است. در موقعیت‌یابی بر اساس روش پیشنهادی RMSE موقعیت مشابه آزمایش قبلی است و کمتر از ۰,۰۸ است. در حالی که RMSE موقعیت در موقعیت یابی بر اساس PF و EKF در این آزمایش افزایش یافته است و بترتیب کمتر ۰,۹ و ۳/۵ شده است.



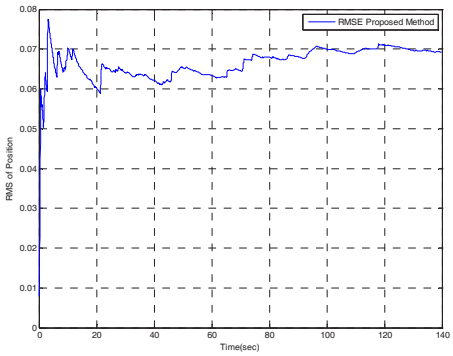
شکل۱۶. مسیر واقعی و تخمین زده شده توسط روش پیشنهادی



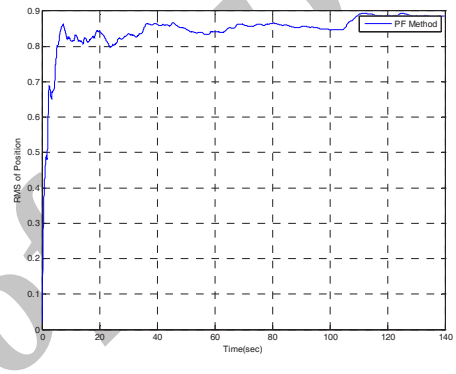
شکل۱۷. مسیر واقعی و تخمین زده شده توسط PF



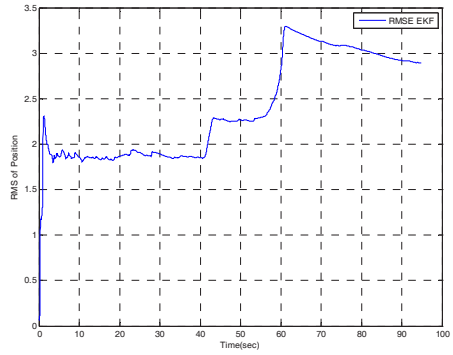
شکل ۱۸ . مسیر واقعی و تخمین زده شده توسط EKF



شکل۱۹. RMSE موقعیت‌یابی براساس روش پیشنهادی



شکل۲۰. RMSE موقعیت‌یابی براساس PF



شکل۲۱. موقعیت‌یابی براساس EKF

نتیجه گیری

در این مقاله، موقعیت یابی بر اساس الگوریتم تکامل تفاضلی ارایه شده است. در روش پیشنهادی، مسئله موقعیت‌یابی تبدیل به یک مسئله بهینه سازی شده و سپس از الگوریم تفاضل تکاملی برای تخمین موقعیت و جهت ربات استفاده شده است. عمکرد موقعیت‌یابی مبتنی بر الگوریتم تکامل تفاضلی با عملکرد موقعیت یابی با فیلتر ذره‌ای و فیلتر کالمن توسعه یافته از نظر دقت و حجم محاسبات مورد ارزیابی قرار گرفته است. نتایج شبیه‌سازی‌ها نشان می‌دهند که عملکرد موقعیت‌یابی بر اساس الگوریتم تکامل تفاضلی بهتر از موقعیت‌یابی بر اساس فیلتر ذره‌ای و فیلتر کالمن توسعه یافته است. یکی از مزایای موقعیت‌یابی بر اساس الگوریتم تکامل تفاضلی در این است که این الگوریتم مانند موقعیت‌یابی براساس فیلتر ذره‌ای نیاز به تابع توزیع پیشنهادی و گام نمونه‌برداری مجدد ندارد. بنابراین خطر تباهیدگی کاهش می‌یابد. علاوه براین، این الگوریتم به تعداد ذرات کمتری برای بدست آوردن دقتی برابر با موقعیت-یابی با فیلتر ذره‌ای نیاز دارد.

مراجع

[1] L.Jetto, S.Longhi, “Development and Experimental Validation of an Adaptive Extended Kalman Filter for the Localization of Mobile Robots”, IEEE Trans. On Robotics And Automation, vol. 15, no. 2, April 1999.

[2] S. Thrun, W. Burgard, and D. Fox, “Probabilistic Robotics”, MIT Press, 2005.

[3] M. Pinto, A.P. Moreira and A. Matos, “Localization of mobile robots using an extended Kalman filter in a LEGO NXT,” IEEE Transactions on Education, vol.55, no.1, pp.135-144,2012.

[4] S.Thrun, “Particle Filters in Robotics”, In Proceedings of Uncertainty in AI (UAI), 2002.

[5] S. Y. Chen, “ Kalman Filter for Robot Vision: A Survey” , IEEE Trans. on Industrial Electronics, pp. 4409-4419 vol. 59, no. 11, November, 2012.

[6] J.Kim, Y.Kim, and S.Kim, “An accuratelocalization for mobile robot using extended kalman filter and sensor fusion”, Proceeding of the International Joint Conference on Neural Networks, 2008.

[7] M.Pinto, A.P. Moreira, A.Matos, “Localization of Mobile Robots Using an Extended Kalman Filter in a LEGO NXT”, IEEE Trans. on Education vol: 55, no.1,pp.135-144,2012.

[8] T.H.Cong, Y.J. Kim and M.T.Lim, “Hybrid Extended Kalman Filter-based Localization with a Highly Accurate Odometry Model of a Mobile Robot”, International Conference on Control, Automation and Systems, 2008

[9] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, “Robust Monte Carlo localization for mobile robots”, Artificial Intelligence, vol. 128, no. 1-2, pp. 99-141, May 2001.

Using Particle Filters", International Conference on Control, Automation and Systems, 2008.
[23] U., Yüzgeç, "Performance comparison of differential evolution techniques on optimization of feeding profile for an industrial scale baker's yeast fermentation process", ISA Transactions, vol.49,no.1,pp.167-176,2010.
[24] G. Liu, Z. Guo, "A clustering-based differential evolution with random-based sampling and Gaussian sampling", Neurocomputing, vol.205, PP. 229-246,2016.
[25] X.Zhou, G.Zhang, X.Hao, L.Yu, "A novel differential evolution algorithm using local abstract convex underestimate strategy for global optimization", Computers & Operations Research, vol.75,pp.132-149,2016.
[26] S.Park, J.Hwang, E.Kim, and H.Kang, "A New Evolutionary Particle Filter for the Prevention of Sample Impoverishment," IEEE Trans. On Evolutionary Computation, vol.13, no.4, August 2009.
[27] B. Ristic, S. Arulampalam, and N. Gordon, in Beyond Kalman Filter:Particle Filters Tracking Applicat, 1st ed. Boston, 2004, ch. 1–3, pp.3–65.
[28] D.Simon, Optimal State Estimation Kalman, H_∞ and Nonlinear Approaches, John Wiley and Sons, Inc, 2006.
[29] Andries P. Engelbrecht, "Computational intelligence: An introduction', Second Edition, Johan Wiley & Sons, Ltd 2007
[30] U.,Yüzgeç, "Performance comparison of differential evolution techniques on optimization of feeding profile for an industrial scale baker's yeast fermentation process", ISA Transactions,2010,vol. 49, no.1 ,pp.167-176 .
[31] Y. ,Bar-Shalom, , X. R. , Li and T. ,Kirubarajan, "Estimation with applications to tracking and navigation , John Wiley and Sons, 2001.
[10] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssel, J. Jansson, R. Karlsson, and P. J. Nordlund, "Particle Filters for positioning, navigation and tracking", IEEE Trans. Signal Processing, vol. 50, no.2, pp. 425-436, 2002.
[11] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssel, J. Jansson, R. Karlsson, and P. J. Nordlund, "Particle Filters for positioning", Navigation and tracking", IEEE Trans. Signal Processing, vol. 50, no.2, pp. 425-436, 2002.
[12] M.Sanjeev Arulampalam, S.Maskell, N.Gordon, and Tim Clapp, "A Tutorial on ParticleFilters for Online Nonlinear/Non-Gaussian Bayesian Tracking", IEEE Trans. on Signal Processing, vol.50, no.2, pp.174–188,2002.
[13] A.A.Wardhana, E.Clearesta, A.Widyotriatmo, S.o, "Mobile Robot Localization Using Modified Particle Filter" , the 3rd International Conference on Instrumentation Control and Automation (ICA), Bali, Indonesia, August 28-30, 2013
[14] B.F.Wu,C.L.Jen, "Particle-Filter-Based Radio Localization for Mobile Robots in the Environments With Low-Density WLAN APs",Trans. Industrial Electronics, vol.61, no.12, pp.6860-6870,2014.
[15] S.Georgios, S.Theodore and T. Anthony, "Intelligent Particle-Filter based Robot Localization", the 19th Mediterranean Conference on Control and Automation Aquis Corfu Holiday Palace, Corfu, Greece June 20-23, 2011
[16] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems",Journal Amer. Statist. Association, vol. 89, pp 278-288, 1994.
[17] J.Zheng-Wei, G.Yuan-Tao, "Novel Adaptive Particle Filters In Robot Localization", Journal of Acta Automatica Sinica, vol.31, no.6, 2005.
[18] Y.Xia, Y.Yang, " Mobile Robot Localization Method Based on Adaptive Particle Filter", C. Xiong et al. (Eds.): ICIRA 2008, Part I, LNAI 5314, pp.963–972, Springer-Verlag Berlin Heidelberg, 2008.
[19] A.A. Wardhana, E. Clearesta, A. Widyotriatmo and S.o, "Mobile robot localization using modified particle filter," the 3rd International Conference on Instrumentation Control and Automation (ICA), Bali, Indonesia, August 28-30, 2013.
[20] D. Zhuo-hua, F. Ming, C. Zi-xing, YU Jin-xia, "An adaptive particle filter for mobile robot fault diagnosis", Journal of Central South University, 2006.
[21] J.Woo, Y.Kim, J.Lee, M.Lim, "Localization of Mobile Robot using Particle Filter",SICE-ICASE International Joint Conference, 2006.
[22] G.Cen, N.Matsuhira, J.Hirokawa, H.Ogawa, I.Hagiwara,"Mobile Robot Global Localization