Indian Journal of Science and Technology, Vol 8(20), DOI:10.17485/ijst/2015/v8i20/52470, August 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Enhanced SLAM for Autonomous Mobile Robots using Unscented Kalman Filter and Neural Network

Omid Panah[1], Amir Panah[2,5*], Amin Panah[3] and Samere Fallahpour[4]

[1]Department of Computer, Ayatollah Amoli Branch, Islamic Azad University, Amol, Iran; o.panah@iauamol.ac.ir
[2]Department of Computer and IT, Hadaf Higher Education Institute, Sari, Iran; amir.panah2020@gmail.com
[3]Department of Computer, Yazd Branch, Islamic Azad University, Yazd, Iran; amin.panah2020@gmail.com
[4]Mazandaran University of Medical Sciences, Sari, Iran; samere.fallahpour@gmail.com
[5]Young Researchers and Elite Club, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Abstract

The novel method of mobile robot Simultaneous Localization And Mapping (SLAM), which is implemented by optimized Unscented Kalman Filter (UKF) Via a Radial Basis Function (RBF) for autonomous robot in unknown indoor environment is proposed. For atone the Unscented Kalman Filter based SLAM errors intrinsically caused by its linearization process, the Radial Basis Function Network is composed with Unscented Kalman Filter. A mobile robot localizes itself autonomously and makes a map simultaneously while it is tracking in an unknown environment. The offered approach has some benefits in handling a robotic system with nonlinear movements because of the learning feature of the Radial Basis Function. The simulation results show the powers and effectiveness of the proposed algorithm comparing with a Standard UKF.

Keywords: Hybrid Filter, Mobile robot, RBF, SLAM, UKF

1. Introduction

A keyword requisite for a correctly autonomous robot is that it can localize itself and carefully map its surroundings simultaneously[1]. Many algorithms have been offered for solving SLAM obstacles, for example particle filter, Kalman Filter (KF), Extended Kalman Filter (EKF) and an Unscented Kalman Filter (UKF). The UKF SLAM making a Gaussian noise supposition for the robot observation and its movement. The UKF encourage calculation values analogous to the ones of EKF but does not requirement the linearization of the fundamental model[2].

The UKF uses the unscented transform to linearize the movement and measurement models[3]. RBF, adaptive to the change of environmental data flowing through the network during the process, can be combined with an UKF to atone for some of the disadvantages of an UKF SLAM method[4].

Amir Panah[5] solved the SLAM problem with a neural network based on an Unscented Kalman filter. According to the research results, the UKF SLAM based on Neural Network, shows better performance than the Standard UKF SLAM.

Stubberud et al.[6] extended an adaptive EKF composed with neural networks, with a neuro-observer to learn system uncertainties on-line. The offered system boost the performance of a control system comprise uncertainties in the state-estimator's model.

In this article, we describe a Hybrid way using RBF network and UKF based SLAM for decline uncertainty in compare to SLAM using Standard UKF. we do, consider the power of UKF based RBF algorithm to handle nonlinear attributes of a mobile robot.

We started by introducing some related algorithms on SLAM are explained in section 2, and the Hybrid algorithm is presented in section 3. The detailed description of the simulation and experimental results is shown in Section 4 and finally in Section 5 summarizes the results and gives an outlook on future research activities.

** Author for correspondence*

2. Related Algorithms for SLAM

2.1 Radial basis Function Neural Network

The idea of Radial Basis Function (RBF) Networks derives from the theory of function approximation. The RBF network is a three-layer feed-forward network. Input layer –input layer has one neuron for each predictor variable. Input neurons standardizes the limited area of the values by reduce the median and dividing by the interquartile range. Then the input neurons require the values to each of the neurons in the hidden layer. Hidden layer –has an unlimited number of neurons that the optimal number is decided by the training process. The out coming value is transmitted to the output layer. Output layer – The value of a hidden layer that come to this layer is multiplied by a weight associated with the neuron and transmitted to the output which adds up the weighted values and presents this sum as the output of the network. That generally uses a linear transfer function for the output units and a nonlinear transfer function (normally the Gaussian function) for the hidden units. The nonlinear transfer function (Gaussian function) is applied to the net input to produce a radial function of the distance. The output units implement a weighted sum of the hidden unit outputs[7]. The structure of Radial Basis Function shows in (Figure 1). two steps of Network training: first, determining the weights from the input to the hidden layer; then, the weights from the hidden layer to the output layer are determined[8].

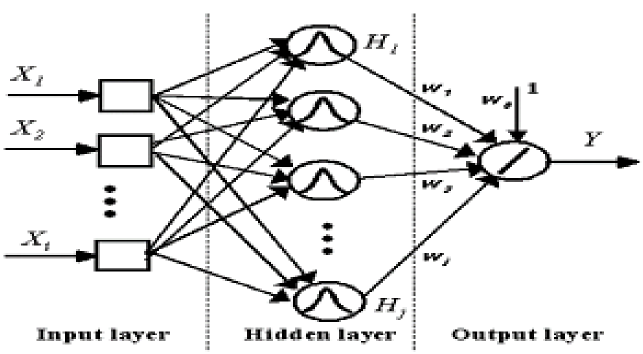

Figure 1. A Radial Basis Function Neural Network Structure.

2.2 Unscented Kalman Filter

UKF introduced by Uhlmann and Julier in 1997 for the first time. This filter structure is based on unscented transformation.

This filter is built based on transformation as unscented transformation. In the UKF, there is no need to calculate Jacobian matrix. Since, the processing noise in this system is accumulative; therefore the augmented state vector is used to implementation this approach. In this approach, the mean and covariance estimation are calculated with considering the second order of the Taylor series.

Assume that a random variable x with covariance P_X and mean μ is defined and also a random variable z as with x is associated: $z=f(x)$. In unscented transformation, to gain the covariance and mean random variable z, sigma points that are set of weighted points are used. This points should be selected so that have a covariance P_X and mean μ.

Calculate the set of 2n+1 sigma points from the columns of the matrix $\sqrt{(n+\lambda)P_X}$:

$$X_0 = \mu, W_0 = \frac{\lambda}{n+\lambda} \tag{1}$$

$$X_i = \mu + \left(\sqrt{(n+\lambda)P_x}\right)_i, W_i = \frac{\lambda}{2(n+\lambda)} \tag{2}$$

$$X_{i+n} = \mu - \left(\sqrt{(n+\lambda)P_x}\right)_i, W_{i+n} = \frac{\lambda}{2(n+\lambda)} \tag{3}$$

$$\lambda = \alpha^2(n+\beta) - n \tag{4}$$

n number of state variables are added. In the above, W_i is the weight which is dependent with the i-th point and k also is used for adjusting the filter more correctly[9].

The UKF makes use of the unscented transform described above to give a Gaussian approximation to the filtering solution of non-linear optimal filtering problems of form, but restated here for convenience:

$$x_k = f\left(x_{k-1}, u_{k-1}, \varepsilon_k\right) \tag{5}$$

$$z_k = h\left(x_k, u_k, \delta_k\right) \tag{6}$$

Where x is state vector and u is control input and ε δ, are the system noise and the measurement noise, respectively. In the first phase of implementing this filter, the augment state vector will become as the following form:

$$X_k^Q = \begin{bmatrix} X_k \\ \varepsilon \\ \delta \end{bmatrix} \tag{7}$$

In continue of this paper, all the formulas that are used in the UKF in which it includes of two main from sections: Measurement update and Time update[10].

- The Time Update

$$X_k^Q = f\left(X_k^Q, u_k, \varepsilon_k\right) \tag{8}$$

$$\mu_k = \sum_{i=0}^{2n} w_i X_{i,k}^Q \tag{9}$$

$$P_k = \sum_{i=0}^{2n} w_i \left[X_{i,k}^Q - \mu_k\right]\left[X_{i,k}^Q - \mu_k\right]^T \tag{10}$$

$$z_k = h(x_k, u_k, \delta_k) \tag{11}$$

$$\bar{z} \quad \sum \quad w_i z_k \tag{12}$$

The Measurement Update

$$P_{x_k x_k} = \sum_{i=0}^{2n} w_i [z_{i,k} - \bar{z}_k][z_{i,k} - \bar{z}_k]^T \tag{13}$$

$$P_{x_k y_k} = \sum_{i=0}^{2n} w_i [X_{i,k}^Q - \mu_k][z_{i,k} - \bar{z}_k]^T \tag{14}$$

$$K_k = P_{x_k y_k} P_{x_k x_k}{}^{-1} \tag{15}$$

$$\mu_k = \mu_k + K_k (z_k - \bar{z}_k) \tag{16}$$

$$P_k = P_k - K_k P_{x_k x_k} K_k{}^T \tag{17}$$

Where $X_k^Q, \mu_k, P_k, z_k, \bar{z}_k, P_{xkxk}, P_{xkyk}$ and K_k, are movement model, predicted mean, observation model, predicted observation, novation covariance, cross correlation matrix and Kalman gain.

3. SLAM Algorithm using Hybrid Filter

A Hybrid filter is proposed in this part, a RBF supplemented for acting as an observer to learn the system completely online. The mean, μ_k which is extracted from values $(x\ y\ \theta\ \varepsilon\ \delta)$ using the RBF algorithm, use for the prediction step, as shown in (Figure 2).

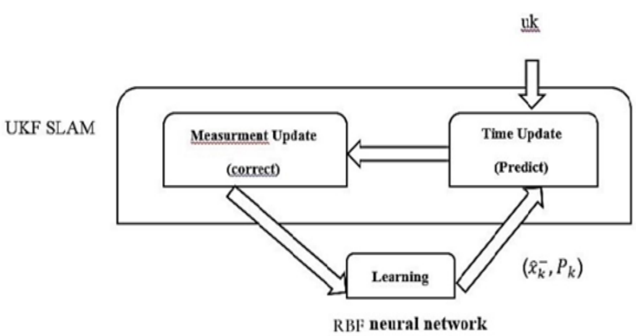

Figure 2. The Architecture of the Hybrid Filter SLAM

Main Basic inputs are covariance, mean which are computed by previous input, u_{k-1}, and present input, u_k. in a prediction step, The robot computes the previous mean and covariance and then, in observation step, it computes a Kalman gain, present mean and covariance and defined features.

RBF neural network can execute as a fast and correct means of approximating a nonlinear mapping based on observed data.

3.1 Time Update (Predict)

The Hybrid filter using landmarks as a robot's position and specifications. A configuration of the robot with a state equation $X^a = (x\ y\ \theta\ \varepsilon\ \delta)^T$, has the form of Equation (18) since it is supposed that the robot is equipped with exteroceptive sensors and encoders.

$$X_k^Q = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ \varepsilon_k \\ \delta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + v_k \Delta t \cos(\theta_k) \\ y_{k-1} + v_k \Delta t \sin(\theta_k) \\ \theta_{k-1} + v_k \Delta t \sin(\frac{\Delta\theta}{L}) \\ \varepsilon_{k-1} \\ \delta_{k-1} \end{bmatrix} \tag{18}$$

$$u_k = v_k + N(0, M_k) \tag{19}$$

v_k is velocity of wheels, L is the width between the robot's wheels, and Δt is the sampling period. Finally, M_k depicts the covariance matrix of the noise in control space. The state equation for landmarks, combined with the robot position. (0<i<c)

$$Y_k^Q = \begin{bmatrix} X_k^Q \\ m \end{bmatrix} = (x_k y_k \theta_k \varepsilon_k \delta_k \quad m_{k,x}^i m_{k,y}^i s_k^i 00)^T \tag{20}$$

The state transition has the form of Eq. (21):

$$X_k^Q = f(X_{k-1}^Q, u_{k-1}) + N(0, \varepsilon_k) \tag{21}$$

ε_k is the process noise, f shows the nonlinear functions, and u_k is control input.

For the Taylor development of function, f its derivative is used with respect to X_k^Q, as shown in Equation (22).

$$f'(X_{k-1}^Q, u_k) = \frac{\partial f(X_{k-1}^u, u_k)}{\partial X_k^Q} \tag{22}$$

f is approximated at u_k and μ_{k-1} The linear extrapolation is achieved by using the gradient of f at u_k and μ_{k-1} as shown in Equation (23).

$$f(X_{k-1}^Q, u_k) = f(\mu_{k-1}, u_k) + f'(\mu_{k-1}, u_k)(X_k^Q - \mu_{k-1}) \tag{23}$$

With the exchange values gained from equations 1, 2, 3, 4, 5, previous covariance and mean have the following form of:

$$\mu_k = \sum_{i=0}^{2n} w_i X_{i,k}^a \tag{24}$$

$$P_k = \sum_{i=0}^{2n} w_i [X_{i,k}^a - \mu_k][X_{i,k}^a - \mu_k]^T \tag{25}$$

As demonstrated in Equation (26), the observation model, z_k contains of the nonlinear measurement function h and the observation noise δ_k.

$$z_k = h\left(Y_k^Q\right) + N(0,\delta_k) = \begin{bmatrix} r_k^i \\ \varnothing_k^i \end{bmatrix} = \begin{bmatrix} \sqrt{\left(m_{k,x}^i - x_k\right)^2 + \left(m_{k,y}^i - y_k\right)^2} \\ \tan^{-1}\left(\dfrac{m_{k,y}^i - y_k}{m_{k,x}^i - x_k}\right) - \theta_k \end{bmatrix} + N(0,\delta_k) \quad (26)$$

$$m^i = \left(m_x^i \quad m_y^i\right)^T \quad (27)$$

$$\bar{z}_k = \sum_{i=0}^{2n} w_i z_k \quad (28)$$

3.2 The Measurement Update (Correct)

To obtain the values P_{xkyk} and P_{xkyk}, it is necessary to compute X_k^Q μ_k $\bar{z}_k$ z_k that are compute in equations 18, 24, 26, 28, with replacement of these values. To compute the Kalman gain K_k, we need to compute P_{xkyk} and P_{xkyk} in the feature-based maps. we will have the following equations:

$$P_{xkxk} = \sum_{i=0}^{2n} w_i [z_{i,k} - \bar{z}_k][z_{i,k} - \bar{z}_k]^T \quad (29)$$

$$P_{xkyk} = \sum_{i=0}^{2n} w_i [X_{i,k}^a - \mu_k][z_{i,k} - \bar{z}_k]^T \quad (30)$$

$$K_k = P_{xkyk} P_{xkxk}^{-1} \quad (31)$$

Considering Hybrid algorithm in continuous. RBF algorithm is included with train through input data and measurement values. In the training process, weights are determined based on the communication of input and each hidden layers. RBF require higher weight to aim value on the higher relations between poses and heading angle with comparing to measurement. In addition, the second weight, ω_0, equals zero because the output offset is zero. Therefore, new estimated mean, can be described as in Equation (32)[11] ($0 \le j \le J-1$)

$$\hat{\mu}_k^j = \omega_0 + \sum_{j=0}^{j-1} \omega_i \varphi_k^j\left(\mu_k^j\right) = \xi\left(\sum_{j=0}^{j-1} \varphi_k^j\left(\mu_k^j\right)\right) = \xi\left(\sum_{j=0}^{j-1} \exp\left(-\frac{\|\mu_k^j - d^j\|^2}{2(\tau^j)^2}\right)\right) \quad (32)$$

d^j is the center of the j-th basis function with the same dimension of the input vector and μ_k is an n-dimensional input vector. τ^i explain the width of the basis function, N is the number of hidden layer's nodes, $\|\mu_k^j - d^j\|$ explains the Euclidean norm of showing the distance between μ_k^j and, d^j and $\varphi_k^j(x)$ means the answer of the j-th basis function of the input vector with a maximum value at d^j.

$$\mu_k = \hat{\mu}_k + K_k\left(z_k - \bar{z}_k\right) \quad (33)$$

$$P_k = P_k - K_k P_{xkxk} K_k^T \quad (34)$$

4. Simulations

To showing the effectiveness of offered algorithm, we used the amend Matlab code that was developed by Bailey[12]. The simulation was carry out by a robot with maximum speeds of 3[m/sec] and wheel diameter of 1[m]. speed and the maximum steering angle are 15[°/sec] and 25[°] respectively.

For observation step, the number of arbitrary features around waypoints was used. In the observation step, a range bearing sensor model and an observation model were used to measure robot pose and the feature position, which includes 1[°] in bearing and a noise with level of 0.1[m] in range. The sensor range is 15[m] for small areas and 25[m] for large areas.

In this paper, two navigation cases of the robot are surveyed: a Circular navigation, and Widespread navigation.

4.1 Navigation on Circular Map

In circular navigation, the standard UKF based navigation and proposed filter based navigation are shown in (Figure 3). Bold black line is Robot path and the dashed line, show the paths of robots should traverse, based on data described by the actual odometry. In (Figure 4), the bold gray line (RBFUKF) and the dashed black line (UKF) are the x, y, and θ errors in the case of UKF SLAM and proposed filter SLAM, and in Table.1 see Mean Square Error of it.

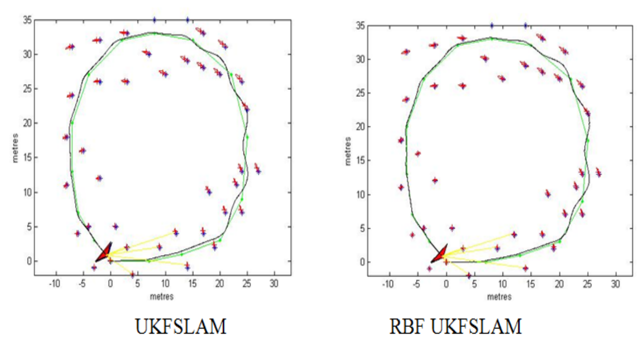

Figure 3. Navigation result on circular map.

Table 1. Mean Square Error

mse	UKFSLAM	RBF UKFSLAM
x	0.1681	0.1119
y	0.1934	0.0583
θ	0.0282	0.0280

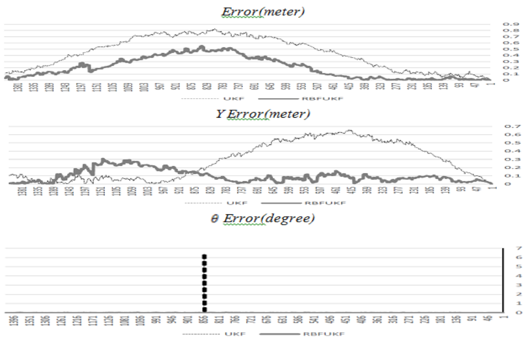

Figure 4. Navigation Errors on Circular Map.

4.2 Navigation on Widespread map

In this case, the UKF based navigation and proposed filter based navigation are shown in (Figure 5). The bold black line is Robot path and the dashed line, show the paths of robots should traverse. (In Figure 6.), the dashed black line (UKF) and the bold gray line (RBFUKF) are the x, y, and θ errors in the case of UKF SLAM and Hybrid filter SLAM with RBF algorithm, respectively and in Table 2. see Mean Square Error of it.

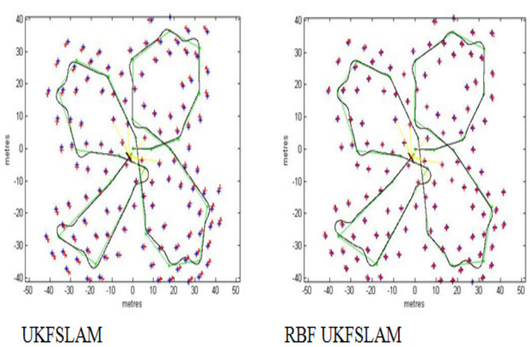

Figure 5. Navigation result on widespread map.

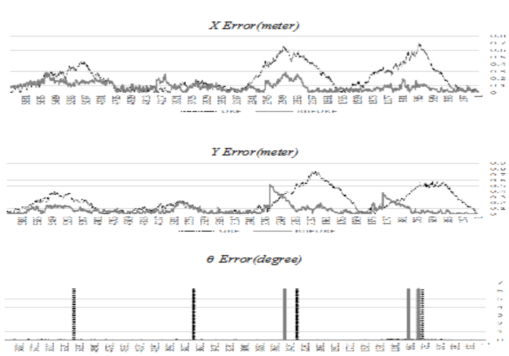

Figure 6. Navigation Errors on Widespread map

Table 2. Mean Square Errorr

mse	*UKFSLAM*	*RBF UKFSLAM*
x	0.0750	0.0112
y	0.0835	0.0135
θ	0.0325	0.0259

5. Conclusion

In this contribution, a new approach proposes UKF SLAM based on RBF Network method for a mobile robot. The RBFUKF SLAM on a mobile robot, to make up for the UKF SLAM error inherently caused by its noise assumption and linearization process. The proposed algorithm contains of two steps: the UKF algorithm and the RBF Neural Network. The simulation results for two different navigation cases show that the improvement of the proposed filter based on RBF as compared with the standard UKF SLAM and it also shown that algorithms has good results in wider environment but we require to use long range sensors. To define the robustness of the proposed algorithm, simulation in Matlab is performed. Based on the simulation results, Standard UKF SLAM has more errors than proposed filter.

6. References

1. Bailey T. Mobile robot localisation and mapping in extensive outdoor environments [degree of Doctor of Philosophy]. Australian Centre for Field Robotics Department of Aerospace, Mechanical and Mechatronic Engineering the University of Sydney; 2002 Aug.
2. Choi K-S, Lee S-G. Enhanced SLAM for a mobile robot using extended kalman filter and neural networks. International Journal of Precision Engineering and Manufacturing. 2010; 11(2):255–64.
3. Frese U. A discussion of simultaneous localization and mapping. Autonomous Robots. Netherlands: Springer Science, Business Media, Inc; 2006.
4. Ranjan TN, Nherakkol A, Navelkar G. Navigation of autonomous underwater vehicle using extended kalman filter. Trends in Intelligent Robotics. 2010. p. 1–9.
5. Panah A, Faez K. hybrid filter based simultaneous localization and mapping for a mobile robot. 16th International Conference on Image Analysis and Processing; 2011 Sep 14; Italy.
6. Chang HH, Lin SY, Chen YC. SLAM for indoor environment using stereo vision. Second WRI Global Congress on Intelligent Systems. 2010.
7. Durrant-Whyte H, Bailey T. Simultaneous localization and mapping: part I. IEEE Robotics & Automation Magazine. 2006; 13(2):99–110.

8. Wolf DF, Sukhatme GS. Mobile robot simultaneous localization and mapping in dynamic environments. Autonomous Robots. 2005; 19:53–65.
9. Ip YL, Rad AB, Wong YK. Effective method for autonomous simultaneous localization and map building in unknown indoor environments. In: Kolski S, editor, Mobile Robots: Perception and Navigation; 2007 Feb. p. 704. ISBN: 3-86611-283-1.
10. Page SF. Multiple-Object sensor Management and optimization [PHD thesis]. Faculty of Engineering, Science and mathematics School of Electronics and Computer science; 2009.
11. Cai X, Zhang Z, Venayagamoorthy G, Wunsch G II. Time series prediction with recurrent neural networks trained by a hybrid pso-ea algorithm. Neurocomputing. 2007;70:2342–53.
12. Bailey T. Available from: http://www.personal.acfr.usyd.edu.au/tbailey