A Neuro-Fuzzy Assisted Extended Kalman Filter-Based Approach for Simultaneous Localization and Mapping (SLAM) Problems

Amitava Chatterjee and Fumitoshi Matsuno, *Member, IEEE*

***Abstract*—Extended Kalman filter (EKF) has been a popular choice to solve simultaneous localization and mapping (SLAM) problems for mobile robots or vehicles. However, the performance of the EKF depends on the correct *a priori* knowledge of process and sensor/measurement noise covariance matrices (Q and R, respectively). Imprecise knowledge of these statistics can cause significant degradation in performance. The present paper proposes the development of a new neurofuzzy based adaptive Kalman filtering algorithm for simultaneous localization and mapping of mobile robots or vehicles, which attempts to estimate the elements of the R matrix of the EKF algorithm, at each sampling instant when a "measurement update" step is carried out. The neuro-fuzzy based supervision for the EKF algorithm is carried out with the aim of reducing the mismatch between the theoretical and the actual covariance of the innovation sequences. The free parameters of the neuro-fuzzy system are learned offline, by employing particle swarm optimization in the training phase, which configures the training problem as a high-dimensional stochastic optimization problem. By employing a mobile robot to localize and simultaneously acquire the map of the environment, under several benchmark environment situations with varying landmarks and under several conditions of wrong knowledge of sensor statistics, the performance of the proposed scheme has been evaluated. It has been successfully demonstrated that in each case, the neuro-fuzzy assistance is able to improve highly unpredictable, degrading performance of the EKF and can provide robust and accurate solutions.**

***Index Terms*—Extended Kalman filter (EKF), neuro-fuzzy assistance, sensor statistics, simultaneous localization and mapping (SLAM) problem.**

I. Introduction

THE simultaneous localization and mapping (SLAM) problem has attracted significant attention from the research communities of the autonomous vehicles and mobile robots in the past two decades. The SLAM problem, essentially, consists of estimating the unknown motion of a moving platform iteratively, in an unknown environment and, hence, determining the map of the environment consisting of features (also known as landmarks) and the absolute location of the moving platform on the basis of each other's information [1]. This is known as a very complex problem as there is always the possibility that both the vehicle's pose estimate and its associated map estimates become increasingly inaccurate in absence of any global position information [2]. This situation arises when a vehicle does not have access to a global positioning system (GPS). Hence the complexity of the SLAM problem is manifold and requires a solution in a high dimensional space due to the mutual dependence of vehicle pose and the map estimates [3].

One of the oldest and popular approaches to solve the SLAM problem employs Kalman filter based techniques. Until now extensive research works have been reported employing EKF to address several aspects of the SLAM problem [1], [4]–[12]. Several successful applications of SLAM algorithms have been developed for indoor applications [13], [14], outdoor applications [7], underwater applications [15], underground applications [16] etc. An EKF based approach, estimates and stores, the robot pose and the feature positions within the map of the environment in the form of a complete state-vector and the uncertainties in these estimates are stored in the form of error covariance matrices. These covariance matrices also include cross-correlation terms signifying cross-correlation among feature/landmark estimates. However, one of the well-known problems with the classical full EKF-based SLAM approach is that the computational burden becomes significantly high in the presence of a large number of features, because both the total state vector and the total covariance matrix become large in size. The later variations of researches on EKF based SLAMs have identified this problem as a key area and several improvements have so far been proposed [7], [9], [17]–[19]. Another key problem associated with EKF-based SLAM is the data association problem, which arises because several landmarks in the map may look similar. In those situations, different data association hypotheses can give rise to multiple, distinct looking maps and Gaussian distribution cannot be employed to represent such multimodal distributions. This problem is usually solved by restricting the algorithm to associate the most likely data association, given the current robot map, on the basis of single measurement [1] or on the basis of multiple measure-

Manuscript received October 15, 2005; revised April 12, 2006, September 27, 2006, and November 8, 2006. The work of A. Chatterjee was supported by the JSPS Postdoctoral Fellowship for Foreign Researchers in Japan. He was on leave from Jadavpur University, Kolkata, India, when this work was performed.

A. Chatterjee was with the University of Electro-Communications, Chofu, Tokyo 182-8585, Japan. He is now with the Electrical Engineering Department, Jadavpur University, Kolkata –700 032, India (e-mail: cha_ami@ yahoo.co.in).

F. Matsuno is with the University of Electro-Communications, Chofu, Tokyo 182-8585, Japan, and also with the International Rescue System Institute, Kawasaki, Kanagawa 210-0855, Japan (e-mail: matsuno@hi.mce.uec.ac.jp).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TFUZZ.2007.894972

ments [20]. The method of utilizing multiple measurements is a more robust method. Although several other data association algorithms have so far been developed, e.g., those in [21] and [22], these algorithms have less significance as they cannot be implemented in real-time.

Some alternative approaches to solve SLAM problems have also been proposed which intend to implement some numerical algorithms, rather than employing the rigorous statistical methods as in EKF. Some of these schemes are based on the Bayesian approaches which can dispense with the important assumption in EKF (i.e., the uncertainties should be modeled by Gaussian distributions). Several such algorithms have been developed employing sequential Monte Carlo (SMC) methods that employ the essence of particle filtering [2], [3], [23], [24]. Particle filtering technique can do away with a basic restriction of EKF algorithm that introduces an additional uncertainty by performing linearization of nonlinear models. However, in particle filtering based methods, it is expected that one should employ large number of particles so that it can contain a particle that can very closely resemble the true pose of the vehicle/robot at each sampling time instant [25]. How to develop an efficient SLAM algorithm, employing particle filtering with small enough number of particles, constitutes an important area of modern-day research. A significant leap in this direction is taken by the FastSLAM1.0 and FastSLAM2.0 algorithms, which have successfully solved the issue of dimensionality for particle filter based SLAM problems [26]. Several other SLAM algorithms have also been successfully developed employing scan-matching technique where the map can efficiently be built by a graph of spatial relations amongst reference frames [7], [27].

It has been shown previously that the performance of an EKF process depends largely on the accuracy of the knowledge of process covariance matrix $(\mathbf{Q})$ and measurement noise covariance matrix $(\mathbf{R})$. An incorrect *a priori* knowledge of $\mathbf{Q}$ and $\mathbf{R}$ may lead to performance degradation [28] and it can even lead to practical divergence [29]. Hence, adaptive estimation of these matrices becomes very important for online deployment. In [28], Mehra has reported a pioneering work on adaptive estimation of noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ for Kalman filtering algorithm, based on correlation-innovations method, that can provide asymptotically normal, unbiased and consistent estimates of $\mathbf{Q}$ and $\mathbf{R}$ [35]. This algorithm is based on the assumption that noise statistics is stationary and the model under consideration is a time invariant one. Later several research works have been reported in the same direction, employing classical approaches, which have attempted adaptive estimation of $\mathbf{Q}$ and $\mathbf{R}$ [30]–[35]. In [30], a combination of an iterative algorithm and a stochastic approximation algorithm has been proposed to estimate $\mathbf{Q}$ and $\mathbf{R}$. In [32] and [33], the problem domain has been expanded to allow time-variance in estimation of $\mathbf{Q}$ and $\mathbf{R}$. A wonderful practical application of [28] has been reported in [34].

In the last ten years or so, there have also been several adaptive Kalman filtering algorithms proposed which employ fuzzy or neuro-fuzzy based techniques [36]–[39]. In [38], an input–output mapping problem, where output is corrupted by measurement noise, is solved by employing a neuro-fuzzy network to determine AR parameters of each operating point dependant ARMA model and then employing Kalman filter for the equivalent state-space representation of the system. In [36], fuzzy logic has been employed for simultaneous adaptive estimations of $\mathbf{Q}$ and $\mathbf{R}$ and in [37], fuzzy logic is employed to adapt the $\mathbf{R}$ matrix only, for a Kalman filter algorithm.

In real world situations, it is quite perceptible that these information matrices, in the form of Q and R, may not be accurately known. Then the performance of the SLAM problem may get affected significantly. The present paper explores those situations for SLAM problems where the noise statistics information for the sensor is not known accurately and proposes to employ a neuro-fuzzy model to assist the EKF-based SLAM algorithm to estimate $\mathbf{R}$ adaptively in each iteration. The free parameters of the neuro-fuzzy model are learned by employing particle swarm optimization (PSO) [40], a stochastic random search based metaheurisitc optimization procedure. Compared to the previously reported fuzzy adapted Kalman filter algorithms, our algorithm proposes a more complicated and sophisticated system than these existing approaches in two main aspects.

i) In each of these previous reports the system under consideration is a simple one where the total number of states is fixed and hence the sizes of the state vector and covariance matrix are always static. For the SLAM problem, the situation is much more complex as the sizes of the state vector and hence the covariance matrix are time varying in nature. This is because, during the process of navigation, new landmarks are initialized in the state vector at different time instants (and, under some specific conditions, some existing landmarks may even be deleted) and hence these vector and matrix sizes will keep changing. The sizes of these matrices usually grow.

ii) The previous reports are also developed using carefully, manually chosen parameters for the fuzzy system(s) under consideration. In contrast, our method proposes a generalized method of learning the neuro-fuzzy model automatically.

The performance of the proposed system is demonstrated by employing it and comparing its performance with a conventional EKF based SLAM algorithm for several environments with varying number of feature/landmark points and with several incorrectly known measurement noise statistics values. These comparisons show that the neuro-fuzzy assistance can indeed improve the degrading performance of the EKF in each test condition significantly.

The rest of the paper is organized as follows. Section II presents a short primer on the already known EKF based SLAM algorithm. Section III presents the proposed neuro-fuzzy assisted EKF based SLAM algorithm. Section IV presents the architecture of the neuro-fuzzy system along with its subsequent training philosophy. In Section V, the performance of the proposed scheme is evaluated. Conclusions are presented in Section VI.

II. EKF-Based Stochastic SLAM Algorithm

A. Hypotheses

- The features under consideration are assumed to be 2-D point features.

- The features are assumed to remain static i.e., they do not change their positions with time, in the map built.
- There are uncertainties in control inputs, the steering angle command (s) and the velocity at which the rear wheel is driven (w), and these uncertainties are modeled using Gaussian distributions.
- It is assumed that there is no uncertainty in the starting pose of the robot.
- The incremental movement of the robot, between two successive sampling instants, is assumed to be linear in nature.
- There are uncertainties in the range (r) and bearing (θ) measurements, and these uncertainties are modeled using Gaussian distributions.
- The features are only characterized by their 2-D positions and no other characteristics, e.g., shape etc., is considered in this work.

B. The Algorithm

A short primer on feature-map based SLAM employing EKF algorithm is presented under this section. A detailed step-by-step description of the algorithm is presented in [6], [7]. An EKF is employed for state estimation in those situations where the process is governed by nonlinear dynamics and/or involves nonlinear measurement relationships. The method employs linearization about the filter's estimated trajectory, which is continuously updated in accordance with the state estimates obtained from the measurements [43]. The state transition can be modeled by a nonlinear function $\mathbf{f}(\bullet)$ and the observation or measurement of the state can be modeled by a nonlinear function $\mathbf{h}(\bullet)$, given as

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{q}_k \tag{1}$$

$$\mathbf{z}_{k+1} = \mathbf{h}(\mathbf{x}_{k+1}) + \mathbf{r}_{k+1} \tag{2}$$

where $\mathbf{x}_k$ is the $(n \times 1)$ process state vector at sampling instant k, $\mathbf{z}_k$ is the $(m \times 1)$ measurement vector at sampling instant k, and $\mathbf{u}_k$ is the control input. The random variables $\mathbf{q}_k$ and $\mathbf{r}_k$ represent Gaussian white process noise and measurement noise, respectively, and $\mathbf{P}_k$, $\mathbf{Q}_k$, and $\mathbf{R}_k$ represent the covariance matrices for $\mathbf{x}_k$, $\mathbf{q}_k$, and $\mathbf{r}_k$, respectively.

In case of the SLAM problem, the state vector $\mathbf{x}$ is composed of the vehicle states $\mathbf{x}_v$ and the landmarks' states $\mathbf{x}_m$. Hence the estimates of the total state vector $\mathbf{x}$, maintained in the form of its mean vector $\hat{\mathbf{x}}$ and the corresponding total error covariance matrix $\mathbf{P}$, is given as

$$\hat{\mathbf{x}} = \begin{bmatrix} \hat{\mathbf{x}}_v^T & \hat{\mathbf{x}}_m^T \end{bmatrix}^T \tag{3}$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_v & \mathbf{P}_{vm} \\ \mathbf{P}_{vm}^T & \mathbf{P}_m \end{bmatrix} \tag{4}$$

where

$\hat{\mathbf{x}}_v$	mean estimate of the robot/vehicle states (represented by its pose);
$\mathbf{P}_v$	error covariance matrix associated with $\hat{\mathbf{x}}_v$;
$\hat{\mathbf{x}}_m$	mean estimate of the feature positions;
$\mathbf{P}_m$	error covariance matrix associated with $\hat{\mathbf{x}}_m$.

The robot/vehicle pose is defined with respect to an arbitrary base Cartesian coordinate frame. The features or landmarks are considered to be 2-D point features. It is assumed that there are n such static, point features observed in the map. Then

$$\hat{\mathbf{x}}_v = \begin{bmatrix} \hat{x}_v & \hat{y}_v & \hat{\varphi}_v \end{bmatrix}^T \tag{5}$$

$$\mathbf{P}_v = \begin{bmatrix} \sigma^2_{x_v x_v} & \sigma^2_{x_v y_v} & \sigma^2_{x_v \varphi_v} \\ \sigma^2_{x_v y_v} & \sigma^2_{y_v y_v} & \sigma^2_{y_v \varphi_v} \\ \sigma^2_{x_v \varphi_v} & \sigma^2_{y_v \varphi_v} & \sigma^2_{\varphi_v \varphi_v} \end{bmatrix} \tag{6}$$

$$\hat{\mathbf{x}}_m = \begin{bmatrix} \hat{x}_1 & \hat{y}_1 & \dots & \hat{x}_n & \hat{y}_n \end{bmatrix}^T \tag{7}$$

and

$$\mathbf{P}_m = \begin{bmatrix} \sigma^2_{x_1 x_1} & \sigma^2_{x_1 y_1} & \cdots & \sigma^2_{x_1 x_n} & \sigma^2_{x_1 y_n} \\ \sigma^2_{x_1 y_1} & \sigma^2_{y_1 y_1} & \cdots & \sigma^2_{y_1 x_n} & \sigma^2_{y_1 y_n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma^2_{x_1 x_n} & \sigma^2_{y_1 x_n} & \cdots & \sigma^2_{x_n x_n} & \sigma^2_{x_n y_n} \\ \sigma^2_{x_1 y_n} & \sigma^2_{y_1 y_n} & \cdots & \sigma^2_{x_n y_n} & \sigma^2_{y_n y_n} \end{bmatrix}. \tag{8}$$

The map is defined in terms of the position estimates of these static features and $\mathbf{P}_{vm}$ in (4) maintains the robot-map correlation. The off-diagonal elements of $\mathbf{P}_m$ signify the cross-correlation and, hence, interdependence of information among the features themselves. The system is initialized assuming that there is no observed feature as yet, the base Cartesian coordinate frame is aligned with the robot's starting pose and there is no uncertainty in the starting pose of the robot. Mathematically speaking, $\hat{\mathbf{x}} = \hat{\mathbf{x}}_v = 0$ and $\mathbf{P} = \mathbf{P}_v = 0$.

As the robot starts moving, $\hat{\mathbf{x}}_v$ and $\mathbf{P}_v$ become non-zero values. In subsequent iterations, when the first observation is carried out, new features are expected to be initialized and $\hat{\mathbf{x}}_m$ and $\mathbf{P}_m$ appear for the first time. This increases the size of $\hat{\mathbf{x}}$ and $\mathbf{P}$ and the entries of $\hat{\mathbf{x}}$ vector and $\mathbf{P}$ matrix are re-calculated. This process is continued iteratively.

1) Time Update ("Predict") Step: Here, it is assumed that the control input vector $\mathbf{u}$, under the influence of which the robot moves, is constituted of two control inputs, the steering angle command (s) and the velocity at which the rear wheel is driven (w). Hence, $\mathbf{u} = \begin{bmatrix} w & s \end{bmatrix}^T$. So, the state estimates can be obtained by employing wheel encoder odometry and the robot kinematic model. The control inputs w and s must be considered with their uncertainties involved (e.g., uncertainties due to wheel slippage, incorrect calibration of vehicle controller) and these are modeled as Gaussian variations in w and s from their nominal values. Hence, the prediction step calculates the projections of the state estimates and the error covariance estimates from sampling instant k to $(k+1)$, given as

$$\hat{\mathbf{x}}^-_{k+1} = \mathbf{f}(\hat{\mathbf{x}}_k, \hat{\mathbf{u}}_k) = \begin{bmatrix} \hat{\mathbf{x}}^-_{v_{k+1}} \\ \hat{\mathbf{x}}_m \end{bmatrix} = \begin{bmatrix} \mathbf{f}_v(\hat{\mathbf{x}}_{v_k}, \hat{\mathbf{u}}_k) \\ \hat{\mathbf{x}}_m \end{bmatrix} \tag{9}$$

$$\mathbf{P}^-_{k+1} = \begin{bmatrix} \nabla \mathbf{f}_{\mathbf{x}_{v_k}} \mathbf{P}_{v_k} \nabla \mathbf{f}^T_{\mathbf{x}_{\mathbf{v}_k}} + \nabla \mathbf{f}_{\mathbf{u}_k} \mathbf{U}_k \nabla \mathbf{f}^T_{u_k} & \nabla \mathbf{f}_{\mathbf{x}_{v_k}} \mathbf{P}_{vm_k} \\ (\nabla \mathbf{f}_{\mathbf{x}_{v_k}} \mathbf{P}_{vm_k})^T & \mathbf{P}_m \end{bmatrix} \tag{10}$$

where $\mathbf{f}_v$ estimates the robot pose on the basis of the motion model and the control inputs. Based on the odometric equation of the mobile robot under consideration here, which assumes

that the incremental movement of the robot is linear in nature, $\mathbf{f}_v$ can be represented as [42]

$$\hat{\mathbf{x}}_{v_{k+1}}^{-} = \begin{bmatrix} \hat{x}_{v_{k+1}}^{-} \\ \hat{y}_{v_{k+1}}^{-} \\ \hat{\varphi}_{v_{k+1}}^{-} \end{bmatrix} = \mathbf{f}_v(\hat{\mathbf{x}}_{v_k}, \hat{\mathbf{u}}_k) = \begin{bmatrix} \hat{x}_{v_k} + w_k * \Delta t * \cos(s_k + \hat{\varphi}_{v_k}) \\ \hat{y}_{v_k} + w_k * \Delta t * \sin(s_k + \hat{\varphi}_{v_k}) \\ \hat{\varphi}_{v_k} + w_k * \Delta t * \frac{\sin(s_k)}{WB} \end{bmatrix} \tag{11}$$

where WB represents the wheelbase of the robot and Δt is the sampling time. The Jacobians and $\mathbf{U}_k$, the covariance matrix of $\mathbf{u}$, are given as

$$\nabla \mathbf{f}_{\mathbf{x}_{v_k}} = \left.\frac{\partial \mathbf{f}_{v_k}}{\partial \mathbf{x}_{v_k}}\right|_{(\hat{\mathbf{x}}_{v_k}, \hat{\mathbf{u}}_k)} \tag{12}$$

$$\nabla \mathbf{f}_{\mathbf{u}_k} = \left.\frac{\partial \mathbf{f}_{v_k}}{\partial \mathbf{u}_k}\right|_{(\hat{\mathbf{x}}_{v_k}, \hat{\mathbf{u}}_k)} \tag{13}$$

$$\mathbf{U} = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_s^2 \end{bmatrix}. \tag{14}$$

Here, $\hat{\mathbf{x}}_m$ and $\mathbf{P}_m$ in (9) and (10) remain constant with time, as the features are assumed to remain stationary with time.

2) Measurement Update ("Correct") Step: Let us assume that the position of an observed feature (say, an i th feature) that already exists in the feature map is denoted by $(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i)$. For the system under consideration [7], [42], it is assumed that the feature observation is carried out using 2-D scanning range laser (SICK PLS), a range-bearing sensor, which nowadays is very popular in mobile robot navigation, for distance measurement. It is assumed that the laser range scanner is mounted on the front bumper of the vehicle and the laser returns a 180° planar sweep of range measurements in 0.5° intervals. The range resolution of such a popular sensor is usually about ±50 mm. In this context, it should be mentioned that the vehicle is also assumed to be equipped with wheel and steering encoders. The distance measured, in polar form, gives the relative distance between each feature and the scanner (and hence the vehicle). Let this feature be measured in terms of its range (r) and bearing (θ) relative to the observer and is given as

$$\mathbf{z} = [\, r \quad \theta \,]^T. \tag{15}$$

The uncertainties in these observations are again modeled by Gaussian variations and let $\mathbf{R}$ be the corresponding observation/measurement noise covariance matrix given as

$$\mathbf{R} = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix} \tag{16}$$

where we assume that there is no cross-correlation between the range and bearing measurements. In connection with the map, the measurements can be given as

$$\hat{\mathbf{z}}_{i_k} = \mathbf{h}_i(\hat{\mathbf{x}}_k) = \begin{bmatrix} \sqrt{(\hat{x}_i - \hat{x}_{v_k})^2 + (\hat{y}_i - \hat{y}_{v_k})^2} \\ \arctan\left(\frac{\hat{y}_i - \hat{y}_{v_k}}{\hat{x}_i - \hat{x}_{v_k}}\right) - \hat{\varphi}_{v_k} \end{bmatrix}. \tag{17}$$

Now, the Kalman gain $\mathbf{W}_i$ can be calculated assuming that there is correct landmark association between $\mathbf{z}$ and $(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i)$ and hence the following computations can be resorted to:

$$\nu_{i_{k+1}} = \mathbf{z}_{k+1} - \mathbf{h}_i\left(\hat{\mathbf{x}}_{k+1}^{-}\right) \tag{18}$$

$$\mathbf{S}_{i_{k+1}} = \nabla \mathbf{h}_{\mathbf{x}_{k+1}} \mathbf{P}_{k+1}^{-} \nabla \mathbf{h}_{\mathbf{x}_{k+1}}^T + \mathbf{R}_k \tag{19}$$

$$\mathbf{W}_{i_{k+1}} = \mathbf{P}_{k+1}^{-} \nabla \mathbf{h}_{\mathbf{x}_{k+1}}^T \mathbf{S}_{i_{k+1}}^{-1} \tag{20}$$

where ν_i denotes the innovation of the observation for the i th landmark and $\mathbf{S}_i$ is the associated innovation covariance matrix. The Jacobian $\nabla \mathbf{h}_{\mathbf{x}_{k+1}}$ is given as:

$$\nabla \mathbf{h}_{\mathbf{x}_{k+1}} = \left.\frac{\partial \mathbf{h}_i}{\partial \mathbf{x}_k}\right|_{\hat{\mathbf{x}}_{k+1}^{-}}. \tag{21}$$

Hence, the *a posteriori* augmented state estimate and the corresponding covariance matrix are updated as

$$\hat{\mathbf{x}}_{k+1}^{+} = \hat{\mathbf{x}}_{k+1}^{-} + \mathbf{W}_{i_{k+1}} \nu_{i_{k+1}} \tag{22}$$

$$\mathbf{P}_{k+1}^{+} = \mathbf{P}_{k+1}^{-} - \mathbf{W}_{i_{k+1}} \mathbf{S}_{i_{k+1}} \mathbf{W}_{i_{k+1}}^T. \tag{23}$$

Here, it should be remembered that in addition to the process and measurement uncertainties, there is an additional uncertainty due to linearization involved in the formulation of an EKF algorithm. The "time update" and "measurement update" equations are obtained by employing linearization of nonlinear functions $\mathbf{f}(\bullet)$ and $\mathbf{h}(\bullet)$ about the point of the state mean. This linearization is carried out by a first order Taylor series expansion and neglecting all the higher terms. This manner of approximating a nonlinear system by a first order derivative introduces this additional source of uncertainty in EKF algorithm. In fact, for highly nonlinear functions, these linearized transformations cannot suffice to provide accurate approximations of the correct covariance transformations and this may lead to highly inconsistent uncertainty estimate. Under those situations unscented transform may provide more accurate results.

3) Initialization of a New Feature and Deletion of an Old Feature: During this iterative procedure of performing prediction and update steps recursively, it is very likely that observations of new features are made time to time. Then these new features should be initialized into the system by incorporating their 2-D position coordinates in the augmented state vector and accordingly modifying the covariance matrix. These features, identified by the LRS, may correspond to points, lines, corners, edges etc. In this work, we have considered that the features are point like features, each representing a unique distinct point in the two-dimensional map of the environment. Resorting to the mathematical computations as shown in [7], these new $\hat{\mathbf{x}}_k^{+}$ and $\mathbf{P}_k^{+}$ can be calculated as

$$\hat{\mathbf{x}}_k^{+} = \begin{bmatrix} \hat{\mathbf{x}}_k \\ \mathbf{f}_f(\hat{\mathbf{x}}_{v_k}, \mathbf{z}_k) \end{bmatrix} \tag{24}$$

$$\mathbf{P}_k^{+} = \begin{bmatrix} \mathbf{P}_{v_k} & \mathbf{P}_{vm_k} & \mathbf{P}_{v_k} \nabla \mathbf{f}_{\mathbf{f}_{v_k}}^T \\ \mathbf{P}_{vm_k}^T & \mathbf{P}_m & \mathbf{P}_{vm_k}^T \nabla \mathbf{f}_{\mathbf{f}_{v_k}}^T \\ \nabla \mathbf{f}_{\mathbf{f}_{v_k}} \mathbf{P}_{v_k} & \nabla \mathbf{f}_{\mathbf{f}_{v_k}} \mathbf{P}_{vm_k} & \nabla \mathbf{f}_{\mathbf{f}_{v_k}} \mathbf{P}_{v_k} \nabla \mathbf{f}_{\mathbf{f}_{v_k}}^T + \nabla \mathbf{f}_{\mathbf{f}_{z_k}} \mathbf{R}_k \nabla \mathbf{f}_{\mathbf{f}_{z_k}}^T \end{bmatrix}. \tag{25}$$

Here, $\mathbf{f}_f(\hat{\mathbf{x}}_v, \mathbf{z})$ is employed to convert the polar observation $\mathbf{z}$ to the base Cartesian coordinate frame. The Jacobians are calculated as

$$\nabla \mathbf{f}_{\mathbf{f}_{v_k}} = \left.\frac{\partial \mathbf{f}_f}{\partial \mathbf{x}_{v_k}}\right|_{(\hat{\mathbf{x}}_{v_k}, \mathbf{z}_k)}$$
$$\nabla \mathbf{f}_{\mathbf{f}_{z_k}} = \left.\frac{\partial \mathbf{f}_f}{\partial \mathbf{z}_k}\right|_{(\hat{\mathbf{x}}_{v_k}, \mathbf{z}_k)}. \quad (26)$$

The deletion of unreliable features is a relatively simpler matter. We only need to delete the relevant row entries from the state vector and the relevant row and column entries from the covariance matrix.

Now, it is quite common that when an observation step is carried out, there will be multiple number of landmarks visible at the same time and hence, several independent observations will be carried out. In our system, we have assumed that a batch of such observations is available at once (i.e., $\mathbf{z} = [r_1, \quad \theta_1, \quad \cdots r_n, \quad \theta_n]^T$) and updates are carried out in batches. This is in conformation with the arguments placed in [7] which indicate that an EKF algorithm tends to perform better update steps for SLAM algorithms, if the innovation vector ν consists of multiple observations simultaneously. Hence, in the context of this batch mode of observation and update procedure, the corresponding SLAM algorithm is based on composite ν, $\mathbf{S}$ and $\mathbf{W}$ vectors/matrices and the sizes of these vectors/matrices keep changing with time because at any instant of observation, the total number of visible landmarks keeps changing.

III. Neuro-Fuzzy Assisted EKF for SLAM Problems

Most of the works reported in the area of adaptive Kalman filters have so far concentrated on utilizing new statistical information from innovation sequence to correct the estimation of the states. Our approach for adapting the EKF is based on the innovation adaptive estimation (IAE) approach, which was originally proposed in [28] and later utilized in combination with fuzzy logic in [37]. The basic concept relies on determining the discrepancy between a new measurement $\mathbf{z}_k$ and its corresponding predicted estimation $\hat{\mathbf{z}}_k$, at any arbitrary k th instant, and utilizing this new information to correct the estimations/predictions already made. The adaptation strategy is based on the objective of reducing mismatch between the theoretical covariance of the innovation sequences $(\mathbf{S}_k)$ and the corresponding actual covariance of the innovation sequences $(\hat{\mathbf{C}}_{Innk})$. In our SLAM algorithm, $\mathbf{S}_k$ is calculated using (19) where the right hand side of the equation is made consistent with the concept of batch mode of observation and update. $\hat{\mathbf{C}}_{Innk}$ can be calculated as

$$\hat{\mathbf{C}}_{Innk} = \nu_k \nu_k^T \quad (27)$$

where ν_k denotes the augmented innovation sequence, made consistent with the batch mode. According to [37], this covariance should be calculated on the basis of a moving average of $\nu_k \nu_k^T$ over an appropriate moving estimation window of size M. However, for the SLAM problem, the size of the augmented ν_k keeps changing from time to time. The reason behind this is it is dependent on the number of landmarks observed in any given *observation and update step*, which have all been observed at least once before. Hence, we use (27) to calculate $\hat{\mathbf{C}}_{Innk}$ rather than using a moving average. Therefore, the mismatch at the k th instant, is given as

$$\Delta \hat{\mathbf{C}}_{Innk} = \hat{\mathbf{C}}_{Innk} - \mathbf{S}_k. \quad (28)$$

Our objective is to minimize this mismatch with the help of fuzzy logic. This is carried out, by employing a one-input–one-output neuro-fuzzy system for each diagonal element of the $\Delta \hat{\mathbf{C}}_{Innk}$ matrix. These fuzzy rules are designed to adapt the $\mathbf{R}$ matrix, so that the sensor statistics is adapted for subsequent reduction in mismatch $\Delta \hat{\mathbf{C}}_{Innk}$. The complete EKF-based SLAM algorithm, assisted by our proposed neuro-fuzzy scheme, is presented in Algorithm 1. The system is designed with a sampling time of 25 msec. between successive control input signals.

The Neuro-Fuzzy Assisted EKF Based SLAM Algorithm.

---

1. ***IF*** All waypoints are traversed, ***THEN*** Stop ***ENDIF***

2. Compute distance of the robot from the current waypoint.

IF (distance $<$ minimum distance allowed from any waypoint),

THEN switch to next waypoint as the current waypoint

ENDIF.

3. Compute change in steering angle (Δs) to point towards the current waypoint and then, new value of steering angle (s) (satisfying the constraints of max. rate of steering change $(\Delta s_{\max})$ and max. steering angle $(s_{\max})$).

4. Move the robot and determine its actual pose.

5. Perform EKF prediction step.

6. ***IF*** (*Time_for_Observation* is TRUE),

THEN go to step 7

ELSE go to step 1

ENDIF.

7. Determine the set of visible landmarks from the current actual robot position. Compute actual range-bearing observation for each of them. Separate those observations based on already observed landmarks and newly observed landmarks (if any).

8. Predict range-bearing observations, for already observed landmarks in step 7, on the basis of augmented total state vector, predicted in step 5.

9. Compute augmented innovation sequence (ν) for already observed landmarks, on the basis of actual and predicted observations, adapted for batch-mode situations.

10. Compute corresponding augmented measurement noise covariance matrix $\mathbf{R}$ (utilizing the original $[2 \times 2]\mathbf{R}$ matrix)

and augmented linearized observation model $\mathbf{h}$, adapted for batch-mode situations.

11. Compute augmented $\mathbf{S}$, on the basis of the augmented $\mathbf{R}$ and $\mathbf{h}$, adapted for batch-mode situations.

12. Update the *a posterior* state estimate vector and error covariance matrix, according to **(22)** and **(23)**.

13. Compute $\hat{\mathbf{C}}_{Innk}$ and $\Delta\hat{\mathbf{C}}_{Innk}$ and determine the size of $\Delta\hat{\mathbf{C}}_{Innk}$, i.e., $[\Delta\hat{\mathbf{C}}_{\text{rows}}, \Delta\hat{\mathbf{C}}_{\text{cols}}]$.

14. Determine the absolute maximum value of mismatch among the range observations $\Delta\hat{\mathbf{C}}_{Innk\_\text{range\_mismatch}}$ and the bearing observations $\Delta\hat{\mathbf{C}}_{Innk\_\text{bearing\_mismatch}}$ separately from the corresponding diagonal entries of the $\Delta\hat{\mathbf{C}}_{Innk}$ matrix.

15. ***FOR*** $j = 1$ to $\Delta\hat{\mathbf{C}}_{\text{rows}}$,

Normalize the corresponding diagonal entry $\Delta\hat{\mathbf{C}}_{Innk}(j,j)$ by the appropriate $\Delta\hat{\mathbf{C}}_{Innk\_\text{range\_mismatch}}$ or $\Delta\hat{\mathbf{C}}_{Innk\_\text{bearing\_mismatch}}$.

Determine the corresponding $\Delta\mathbf{R}(j,j)$ output from the

NFS, with the normalized $\Delta\hat{\mathbf{C}}_{Innk}(j,j)$ input to it.

ENDFOR

16. Determine $\Delta\sigma_r^2$ as a mean of those $\Delta\mathbf{R}(j,j)$ entries, which correspond to range measurements.

17. Determine $\Delta\sigma_\theta^2$ as a mean of those $\Delta\mathbf{R}(j,j)$ entries, which correspond to bearing measurements.

18. Adapt the original 2×2 $\mathbf{R}$ matrix as: $\mathbf{R}_k = \mathbf{R}_{k-1} + \begin{bmatrix} \Delta\sigma_r^2 & 0 \\ 0 & \Delta\sigma_\theta^2 \end{bmatrix}$.

19. ***IF*** (new feature(s) observed in step 7),

THEN augment state vector and error covariance matrix, according to **(24)**, **(25)** and **(26)**.

ENDIF

20. Go to step 1.

From algo. 1, it can be seen that each neuro-fuzzy system (NFS) employs a nonlinear mapping of the form: $\Delta\mathbf{R}(j,j) = f_{\text{NFS}}(\Delta\hat{\mathbf{C}}_{Innk}(j,j))$ where $\Delta\mathbf{R}(j,j)$ corresponds to an adaptation recommended for the corresponding diagonal element of the augmented measurement noise covariance matrix, $\mathbf{R}$, computed according to the batch-mode situation. This augmented matrix is calculated each time an iteration enters into the *observe and update step* and its size is determined on the basis of the total landmarks visible in the *observe step*. To make it consistent with the batch of observed landmarks that has already been visited at least once earlier, the size of this augmented $\mathbf{R}$ is $[2z_f \times 2z_f]$ where z_f is the number of landmarks observed in that iteration, which were also observed earlier. This augmented $\mathbf{R}$ is formed utilizing the original $\mathbf{R}$ matrix having a dimension of 2×2, and this is formulated as:

$$\text{augmented } \mathbf{R} = \begin{bmatrix} \sigma_r^2 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_\theta^2 & \cdots & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & \sigma_r^2 & 0 & \cdots & \vdots & \vdots \\ \vdots & \vdots & 0 & \sigma_\theta^2 & \cdots & \vdots & \vdots \\ \vdots & \vdots & \cdots & \cdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & \cdots & \cdots & \sigma_r^2 & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & \sigma_\theta^2 \end{bmatrix}. \tag{29}$$

Here, σ_r^2 and σ_θ^2 correspond to the sensor statistics computed for that iteration. It can be seen that the augmented $\mathbf{R}$ matrix comprises of diagonal elements only and all the off-diagonal elements are considered to be zero. This is in conformation with our assumptions presented beforehand, in Section II, that the range and the bearing measurements are independent of each other and there is no cross-correlation between these measurements. The size of this augmented $\mathbf{R}$ matrix keeps changing in different iterations, as the number of already visited landmarks, observed again in a given iteration, keeps varying from iteration to iteration. The size of this augmented $\mathbf{R}$ is consistent with that of the $\hat{\mathbf{C}}_{Innk}$ and, hence, $\Delta\hat{\mathbf{C}}_{Innk}$.

With the idea of implementing the same NFS for each and every diagonal element of the augmented $\mathbf{R}$ matrix, we employ normalized input for each NFS. The NFS is realized by three fuzzy ***IF–THEN*** rules of the form

$$\text{IF } \Delta\hat{\mathbf{C}}_{Innk}(j,j) \text{ is } N \text{ THEN } \Delta\mathbf{R}(j,j) = w_1$$
$$\text{IF } \Delta\hat{\mathbf{C}}_{Innk}(j,j) \text{ is } Z \text{ THEN } \Delta\mathbf{R}(j,j) = w_2$$
$$\text{and}$$
$$\text{IF } \Delta\hat{\mathbf{C}}_{Innk}(j,j) \text{ is } P \text{ THEN } \Delta\mathbf{R}(j,j) = w_3.$$

w_1, w_2, and w_3 indicate the amount of fuzzy adaptation recommended in form of a diagonal element of the $\Delta\mathbf{R}$ matrix, depending on the nature of the fuzzified mismatch in the corresponding diagonal element of the $\Delta\hat{\mathbf{C}}_{Innk}$ matrix. However, the order of the mismatch may be different for range and bearing observations and this may depend on how poorly (or accurately) the sensor statistics for range and bearing observations are individually known. Hence we employ normalized inputs corresponding to range and bearing observations separately, on the basis of appropriate computations of $\Delta\hat{\mathbf{C}}_{Innk\_\text{range\_mismatch}}$ and $\Delta\hat{\mathbf{C}}_{Innk\_\text{bearing\_mismatch}}$, as given in algo. 1. Then with these normalized inputs, the NFS enables us to compute $\Delta\mathbf{R}(j,j)$ for each diagonal entry. Finally we compute the adaptations i.e., $\Delta\sigma_r^2$ and $\Delta\sigma_\theta^2$ required for the original $[2\times2]\mathbf{R}$ matrix on the basis of appropriate means, separately computed from the arrays of $\Delta\hat{\mathbf{C}}_{Innk}(j,j)$ entries for range and bearing observations. This adapted original $[2 \times 2]\mathbf{R}$ matrix is kept ready for the next appropriate iteration, when EKF will enter the *observation and update step*, and will be utilized for subsequent formation of augmented $\mathbf{R}$ matrix and so on. Then, each *observation and update step* is concluded by augmenting the state vector and the corresponding covariance matrix, by employing (24)–(26), if there are new feature(s) observed during this *observation step*.

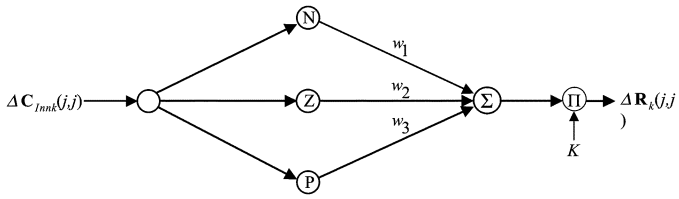


Fig. 1. Four-layer architecture of the proposed neuro-fuzzy system.

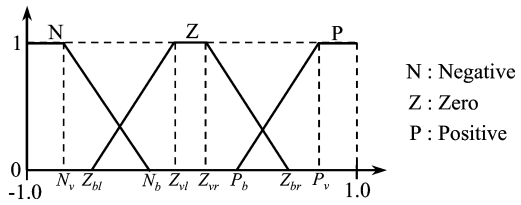


Fig. 2. Membership functions employed in Fig. 1.

IV. The Neuro-fuzzy Architecture and Its Training Methodology

A. Architecture of the Neuro-Fuzzy Model

The neuro-fuzzy model has been developed as an one-input-one-output system. The four-layer architecture is shown in Fig. 1. Let u_i^l and O_i^l denote the input to and output from the i th node of the lth layer, respectively.

a) Layer 1: Input Layer
This layer comprises a single node, signifying the single input variable. The input–output relation of this node is

$$O^1 = u^1 = \Delta\hat{\mathbf{C}}_{Innk}(j,j). \tag{30}$$

b) Layer 2: Membership Function Layer
Here, the input variable is fuzzified employing three Membership Functions (MFs), negative (N), zero (Z) and positive (P). Fig. 2 shows these MFs where N_v and N_b respectively denote the right vertex and right base points of the MF N. Z_{bl}, Z_{vl}, Z_{vr} and Z_{br} respectively denote the left base, left vertex, right vertex and right base points of the MF Z. P_b and P_v respectively denote the left base and left vertex points of the MF P. The output of the i th MF is given as:

$$O_i^2 = \mu_i(u^1) = \mu_i(\Delta\hat{\mathbf{C}}_{Innk}(j,j)). \tag{31}$$

c) Layer 3: Defuzzification Layer
This layer performs defuzzification where the defuzzified output is calculated as an weighted average of all its inputs. Hence the output from the solitary node in this layer can be calculated as:

$$O^3 = \frac{\sum_{i=1}^{3} O_i^2 * w_i}{\sum_{i=1}^{3} O_i^2} = \frac{\sum_{i=1}^{3} \mu_i(u^1) * w_i}{\sum_{i=1}^{3} \mu_i(u^1)}. \tag{32}$$

d) Layer 4: Output Layer
This layer performs a suitable scaling for the defuzzified output. The input–output relationship of the node in this layer is given as

$$O^4 = K * u^4 = K * O^3. \tag{33}$$

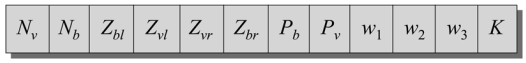


Fig. 3. Detailed configuration of each 12-dimensional "particle" employed by PSO.

B. Training the Neuro-Fuzzy Model Employing Particle Swarm Optimization (PSO)

This neuro-fuzzy model is trained to determine the suitable free parameters of the system i.e., the parameters of the MFs, the output consequence singletons and the output gain. However, the training cannot be accomplished in the conventional supervised mode, as the exact desired output, for a given input, is not quantitatively known. Hence, normal backpropagation kind of training methodology can not be resorted to and it is suitable to apply stochastic global optimization algorithms for such systems in an unsupervised manner. There are several such candidate algorithms available now. In the present work, PSO has been chosen, which is a relatively new algorithm [40], [41], that is based on the swarm behaviors of birds or fishes. The training of the neuro-fuzzy system is accomplished as a high-dimensional metaheuristic optimization problem, where the objective is to optimize a fitness function $f_{fit}(x_1, x_2, \ldots x_n)$ on the basis of the values of the variables $x_1, x_2, \ldots x_n$.

In a PSO problem, several such candidate solutions of $x_1, x_2, \ldots x_n$ are created in a multidimensional space (called "particles") and the suitability of each of them is evaluated in each iteration. For the problem under consideration here, each such potential "particle" is formed as a 12-dimensional vector $\mathbf{x} = [\, x_1 \quad x_2 \quad \cdots \quad x_{12} \,]$, as shown in Fig. 3. Each "particle" i is characterized by the vectors denoting its position $(\mathbf{x}_i)$ and its velocity $(\mathbf{v}_i)$ at the current time step. In order to pursue the optimum of the fitness function (f_{fit}), velocity $\mathbf{v}_i$ and hence position $\mathbf{x}_i$ of each particle is adjusted in each time step. The updated velocity in each time step $\mathbf{v}_{\text{inew}}$ is a function of three major components: the old velocity vector of the same particle $(\mathbf{v}_{\text{iold}})$, difference of the i th particle's best position found so far (called $\mathbf{p}_i$) and the current position $(\mathbf{x}_i)$ (called the "cognitive" component) and difference of the best position of any particle within the purview of the topological neighborhood of i th particle found so far (called $\mathbf{p}_g$) and current position of the ith particle $(\mathbf{x}_i)$ (called the "social" component) [40], [41]. Each of the last two components is stochastically weighted so that the updating in the velocity of each particle will cause enough oscillations, allowing each particle to search for a better pattern within the problem space. Hence, the velocity and position update relations, in the dth dimension, are given as

$$\begin{aligned}
\nu_{\text{idnew}} &= \nu_{\text{idold}} + \phi_i(p_{id} - x_{id}) + \phi_g(p_{gd} - x_{gd}) \\
&\quad \mathbf{IF}(\nu_{\text{idnew}} > \nu_{\text{dmax}})\mathbf{THEN}\ \nu_{\text{idnew}} = \nu_{\text{dmax}} \\
&\quad \mathbf{IF}(\nu_{\text{idnew}} < -\nu_{\text{dmax}})\mathbf{THEN}\ \nu_{\text{idnew}} = -\nu_{\text{dmax}} \\
x_{\text{idnew}} &= x_{\text{idold}} + \nu_{\text{idnew}} \\
\nu_{\text{idold}} &= \nu_{\text{idnew}} \\
x_{\text{idold}} &= x_{\text{idnew}}
\end{aligned} \tag{34}$$

φ_i and φ_g are responsible for introducing stochastic weighting and they are given as $\varphi_i = c_i * \mathrm{rand}_1()$ and $\varphi_g = c_g * \mathrm{rand}_2()$. $\mathrm{rand}_1()$, and $\mathrm{rand}_2()$ are two random functions in $[0, 1]$ and c_i and c_g are positive constants. A popular choice for c_i and c_g is $c_i = c_g = 2$. This traditional PSO model shows quick, aggressive convergence during the early phase but often encounters problem in fine tuning the search to determine the supreme solution. Hence, in our algorithm, we have employed an improved version of this PSO algorithm that utilizes a judicious mix of aggressive, coarse updating during early iterations and fine updating during later iterations [40]. Hence, the velocity update rule is given as

$$\nu_{\mathrm{idnew}} = w_{\mathrm{iter}}(\nu_{\mathrm{idold}}) + \varphi_i(p_{id} - x_{id}) + \varphi_g(p_{gd} - x_{id}) \quad (35)$$

with the position update rule remaining unchanged as given before. w is called the inertia weight which is initially kept high and then is gradually decreased as the iterations progress. Hence, inertia weight initially updates the velocity in a coarse manner and ultimately, with time, fine changes in velocity updating takes over. In our algorithm, we have utilized linearly adaptable inertia weight and w_{iter} gives the value of the inertia weight at that given iteration. The iterative process is continued until the optimization process yields a satisfactory result. This is evaluated on the basis of whether the value of f_{fit} falls below the specified maximum allowable value or whether the maximum number of iterations has been reached. A detailed description of the PSO algorithm is available in [40], [41].

In our approach, the objective of the neuro-fuzzy assistance to the EKF based SLAM is to improve the performance of the estimation as much as possible. This implies that we should try and minimize the discrepancy between the actual covariance and the theoretical covariance of the innovation sequence over the entire set of observation instants, during the movement of the vehicle/robot, as far as possible. Hence the fitness function is formulated on the basis of: a) computing the mean-square value of all the diagonal entries of the $\Delta\mathbf{C}_{Innk}$ matrix, at any given observation instant, b) storing such mean-square values for each observation instant during an ongoing iteration, and c) computing a mean of all such mean-square values for all observation instants at the end of a complete iteration. Mathematically this can be shown as

$$\mathbf{f}_{fit} = \frac{\sum_{n_{\mathrm{obs}}=1}^{N_{\mathrm{obs}}} \left(\frac{\sum_{j=1}^{J_{C\_nobs}} [\Delta C_{Innk}(j,j)]^2}{J_{C\_nobs}} \right)}{N_{\mathrm{obs}}} \quad (36)$$

where N_{obs} denotes the total number of observation instants in a given iteration and J_{C_nobs} denotes the total number of diagonal elements of $\Delta\mathbf{C}_{Innk}$ matrix when the n_{obs} th observation is made.

In the context of adapting a meaningful NFS, the positions of each "particle", at the end of each iteration, are subjected to several constraints. Most of these constraints are implemented to maintain specific shapes chosen for the MFs (usually trapezoidal, which, as a special case, can become triangular) and also to ensure that there is some overlapping between the stretches of consecutive MFs. Another constraint included is that, for each MF, its control points (starting from left to right) should be chosen in a monotonically nondecreasing fashion. This will ensure that all the regions, within the universe of discourse of the input for the NFS, will remain covered by at least one MF. These constraints are implemented as:

$$\begin{aligned}
\mathbf{IF}(N_b < N_v)\mathbf{THEN}\ N_b = N_v\ \mathbf{ENDIF}\\
\mathbf{IF}(Z_{vl} < Z_{bl})\mathbf{THEN}\ Z_{vl} = Z_{bl}\ \mathbf{ENDIF}\\
\mathbf{IF}(Z_{vr} < Z_{vl})\mathbf{THEN}\ Z_{vr} = Z_{vl}\ \mathbf{ENDIF}\\
\mathbf{IF}(Z_{br} < Z_{vr})\mathbf{THEN}\ Z_{br} = Z_{vr}\ \mathbf{ENDIF}\\
\mathbf{IF}(P_v < P_b)\mathbf{THEN}\ P_v = P_b\ \mathbf{ENDIF}\\
\mathbf{IF}(N_b < Z_{bl})\mathbf{THEN}\ N_b = Z_{bl}\ \mathbf{ENDIF}\\
\mathbf{IF}(Z_{br} < P_b)\mathbf{THEN}\ Z_{br} = P_b\ \mathbf{ENDIF}. \quad (37)
\end{aligned}$$

Another constraint is implemented to signify that the scaling employed in the output layer of the NFS is carried out only for magnitude scaling, and hence it cannot be employed for changing polarity. It means that K can never be negative.

V. Performance Evaluation

To evaluate the performance of the proposed system, we have considered various environments, which are available in [42]. In fact, the packages available in [42] should serve as an excellent platform for learning and analysis of existing Kalman filter and particle filter based SLAM algorithms. Researchers can develop their own algorithms and can compare their performance *vis-à-vis* these algorithms. Several benchmark environments are available there and we have tested our algorithm in these simulated environments with their associated given vehicle motion model. The environment is usually specified in such a manner where a vehicle/robot is supposed to navigate through some waypoints and in the process should be able to acquire the map of the environment with several configurations of feature/landmark points. In the present scheme, we consider three such environments as specified in [42]. In each case we have the identical scene of ideal robot movement where the robot path is specified by 17 waypoints. However, each environment consists of varied number of landmarks to impose several degrees of complexities and the three environments, under consideration, consist of 35, 135, and 497 landmarks, respectively. The uncertainties in control inputs are specified as: $\sigma_w = 0.3\ \mathrm{m/sec.}$ and $\sigma_s = 3\ \mathrm{deg.}$ An observation step and the associated update step is carried out after eight consecutive prediction steps, identical with the EKF based algorithm in [42]. This follows a popular notion in EKF-based SLAM community, where instead of employing an observation and update step after each prediction step, one computes several consecutive prediction steps, and then takes corrective action by one observation and update step. This helps in reducing the computational burden of the SLAM algorithm. In algo. 1, this is indicated by the *Time_for_Observation* flag, which is set TRUE for one iteration, after each 8 successive iterations.

The performance of the proposed system is compared with a conventional EKF-based SLAM system where the $\mathbf{Q}$ and $\mathbf{R}$ matrices are kept static throughout the experiment. The proposed

algorithm starts with the same $\mathbf{Q}$ and $\mathbf{R}$ matrices, but it keeps adapting the $\mathbf{R}$ matrix according to the proposed scheme. According to the data available from [42], the EKF based algorithm works perfectly when sensor statistics are known as: $\sigma_r = 0.1$ m. and $\sigma_\theta = 1$ deg. First we consider the situation where the sensor statistics are wrongly considered as: $\sigma_r = 2.0$ m. and $\sigma_\theta = 0.1$ deg. In each figure, the dotted line shown in black, depict the actual path traversed by the robot, while the solid line shown in black, depict the SLAM estimated path traversed based on estimated states of robot poses in each sampling instant or iteration. The stars $(*)$ depict the actual landmark positions, which are stationary in the environment. The crosses $(+)$ depict the positions of these landmarks estimated at the end of the test run. Obviously, the performance of the system will be superior, if the estimated robot path and the actual path match as closely as possible and the estimated landmark positions and their actual positions coincide as closely as possible. Fig. 4(a)–(c) show the performance of the conventional EKF-based SLAM for three different environment situations. It can be seen that the performance is acceptable when there are small number of landmarks in the environment. However, the performance is really worse when the landmarks become denser and both the estimations of the robot pose at different instants and the map acquired, degrade significantly as the EKF estimations are quite distant from the original robot positions and the map situation. Fig. 5(a)–(c) show the situations, when the neuro-fuzzy assisted EKF-based SLAM is employed for identical environments. It can be found that the neuro-fuzzy assistance could improve the situation dramatically and the estimates of the robot states as well as acquisition of the map are quite stable for all three different environments with varied number of landmarks. In all these environments, the robot position estimates follow the actual robot positions closely and the estimations of the stationary landmark positions also closely match with their actual positions in the environments.

The scheme is further tested for another situation where the sensor statistics are wrongly considered in opposite directions and they are considered as $\sigma_r = 0.01$ m and $\sigma_\theta = 3.0$ deg. Then the same set of algorithms is deployed for identical set of environments. Fig. 6(a)–(c) show the performances of the conventional EKF-based SLAM and Fig. 7(a)–(c) show the corresponding performances of the neuro-fuzzy assisted EKF-based SLAM algorithms. In these case studies, the EKF-based SLAM shows a different trend in performance. As we can see, the estimation performance is worst for the environment containing small number of landmarks. However, with increase in landmarks, the estimations become more accurate and for the situation with 497 landmarks, the performance of the EKF-based SLAM was quite satisfactory. On the other hand, the neuro-fuzzy assisted EKF shows uniformly stable performance for each environment with quite accurate estimations of robot poses and feature positions for each environment situation. Each result, shown in Figs. 5(a)–(c) and 7(a)–(c), for the neuro-fuzzy assisted EKF based SLAM, depicts one sample run conducted. For each of these six specific situations of two case studies, we conducted 10 individual runs. It is found that, for each given situation, results obtained with each of these 10 individual runs, are very close to each other. Fig. 8(a)–(c) show

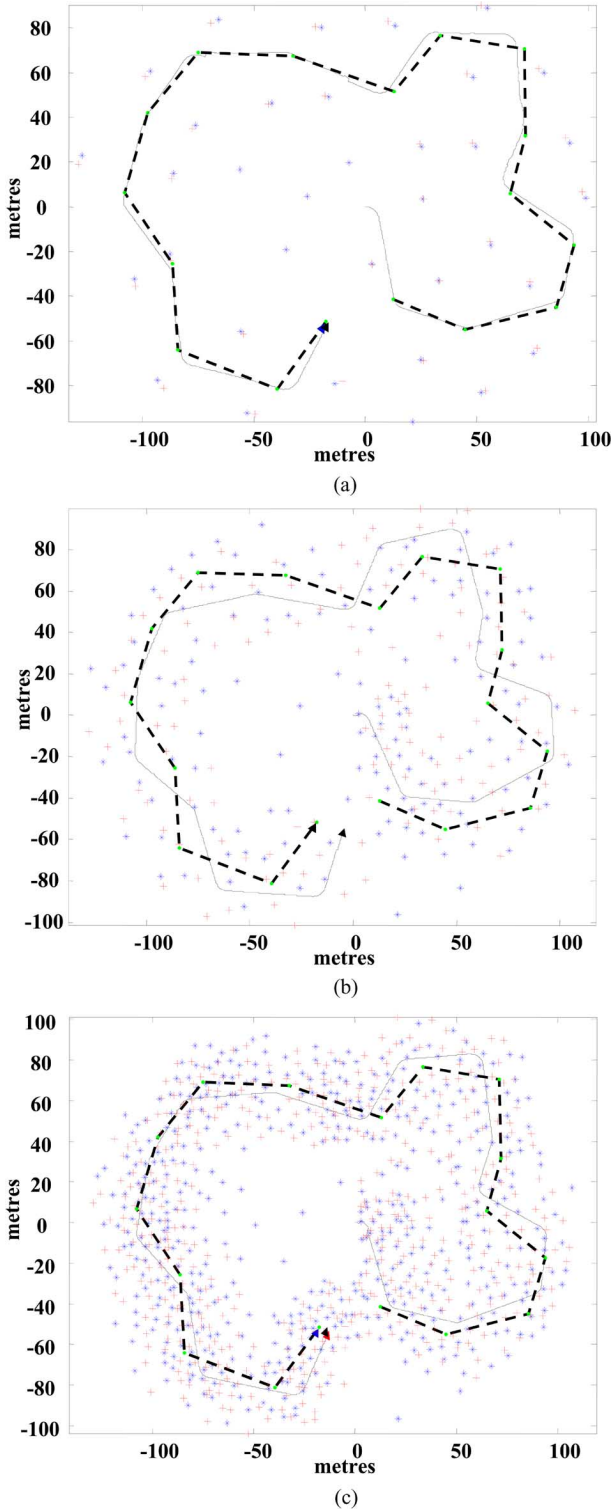


Fig. 4. Conventional EKF-based SLAM performance for case study I ($\sigma_r = 2.0$ m. and $\sigma_b = 0.1$ deg.) with (a) 35, (b) 135, and (c) 497 features/landmarks in the environment.

comparisons of the RMS errors (rmse) in estimating $\hat{\mathbf{x}}$, employing both the EKF-based SLAM only and the neuro-fuzzy assisted EKF-based SLAM, for each incremental movement of the robot, for the case study I. For the neuro-fuzzy assisted EKF-based SLAM, the errors are plotted on the basis of the averages calculated for RMS errors at each sampling instant, computed over 10 individual sample runs. Fig. 9(a)–(c) show

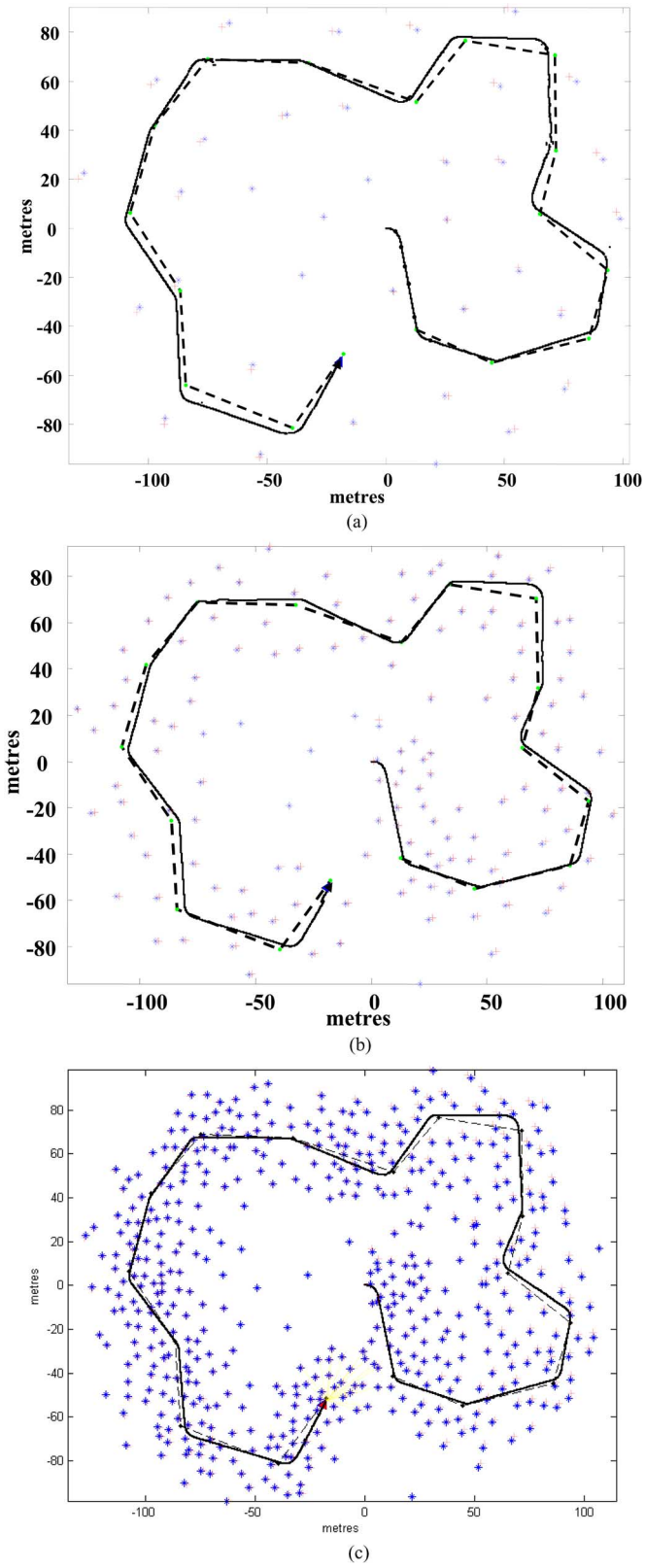


Fig. 5. Neuro-fuzzy assisted EKF-based SLAM performance for case study I ($\sigma_r = 2.0$ m. and $\sigma_b = 0.1$ deg.) with (a) 35, (b) 135, and (c) 497 features/landmarks in the environment.

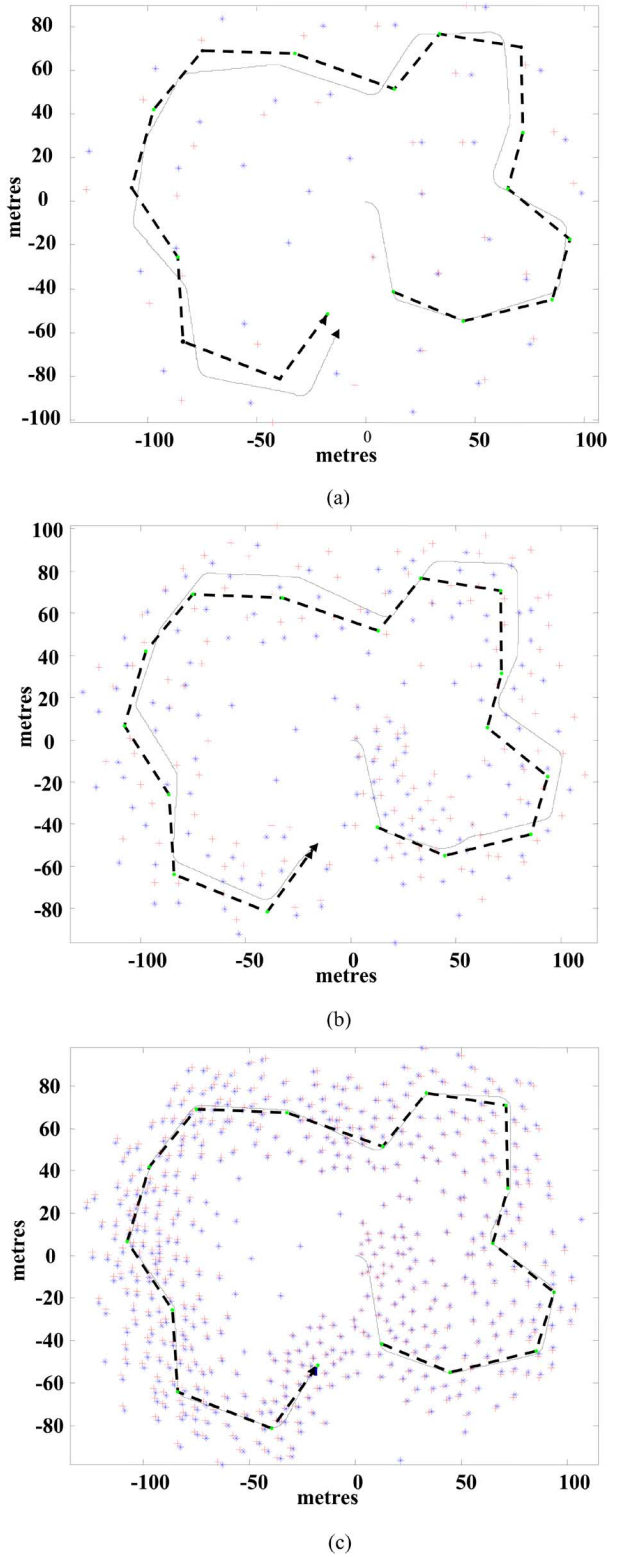


Fig. 6. Conventional EKF-based SLAM performance for case study II ($\sigma_r = 0.01$ m. and $\sigma_b = 3.0$ deg.) with (a) 35, (b) 135, and (c) 497 features/landmarks in the environment.

similar comparisons of RMS errors, for the two algorithms considered, for the case study II. In most of these six situations, our proposed algorithm could produce much lesser magnitudes of rmse compared to the EKF based algorithm. Only for case study I with 17 landmarks, both algorithms show very similar variations of rmse, with EKF-based algorithm showing slightly better performance. Hence, these case studies further prove that the neuro-fuzzy assistance can vastly improve the degrading

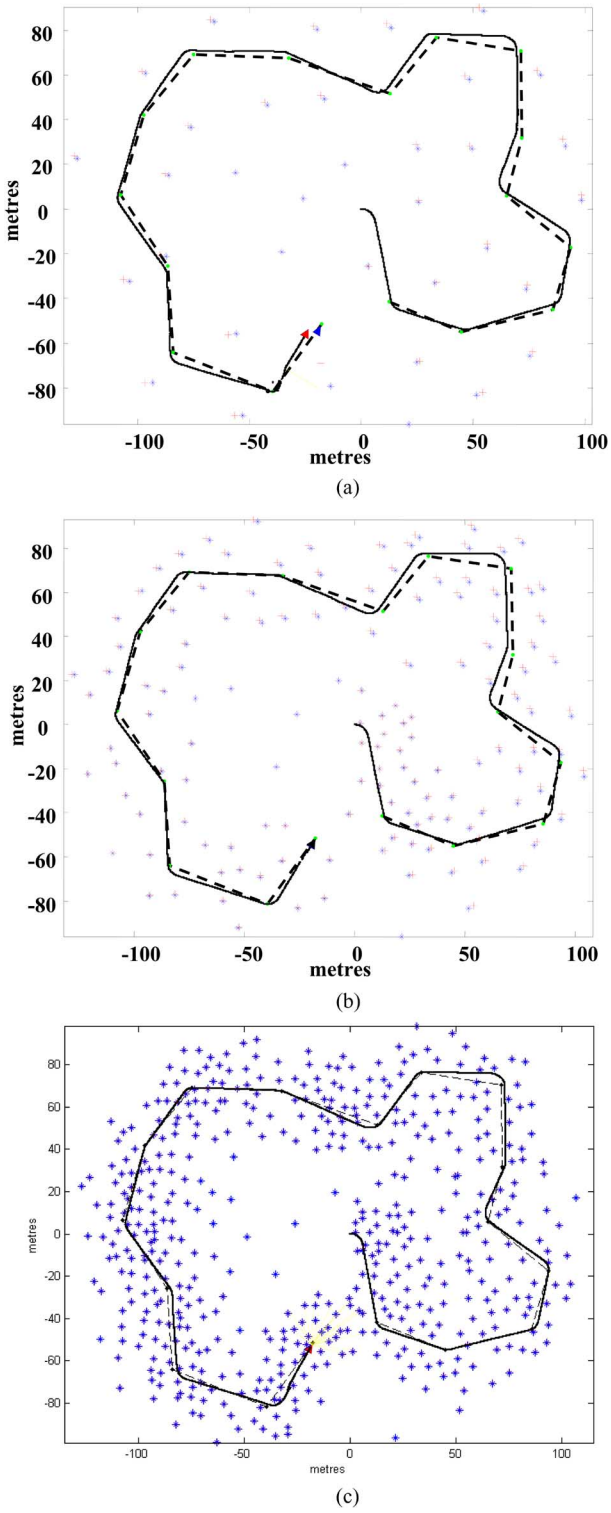


Fig. 7. Neuro-fuzzy assisted EKF-based SLAM performance for case study II ($\sigma_r = 0.01$ m. and $\sigma_b = 3.0$ deg.) with (a) 35, (b) 135, and (c) 497 features/landmarks in the environment.

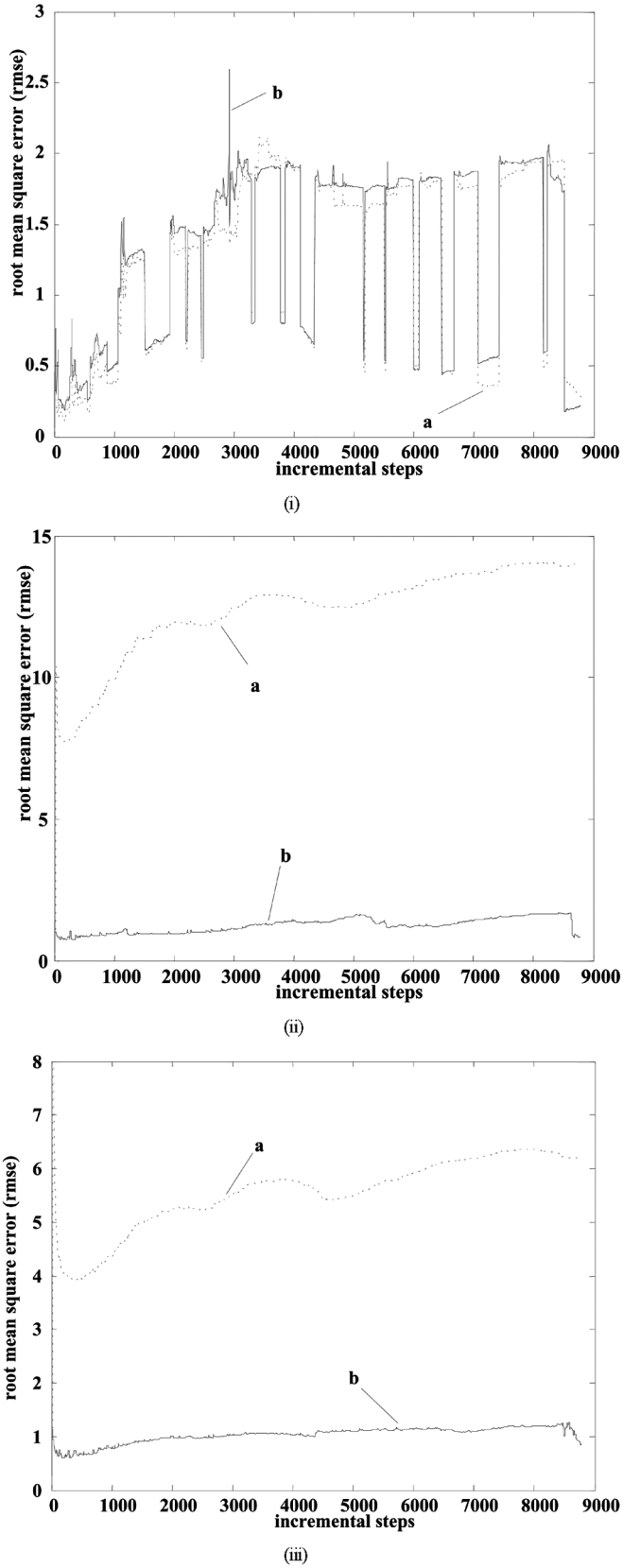


Fig. 8. Comparison of root mean square error for (a) conventional EKF-based SLAM and (b) neuro-fuzzy assisted EKF-based SLAM for case study I ($\sigma_r = 2.0$ m. and $\sigma_b = 0.1$ deg.) with (i) 35, (ii) 135, and (iii) 497 features/landmarks in the environment.

performance of the traditional EKF algorithm in several situations, when the sensor statistics are wrongly known. In these situations, the performance of the conventional EKF becomes highly unreliable. However, presence of neuro-fuzzy assistance can help the EKF to maintain a stable performance and this performance has been shown robust enough over several environment situations, with several wrong knowledge of sensor statistics.

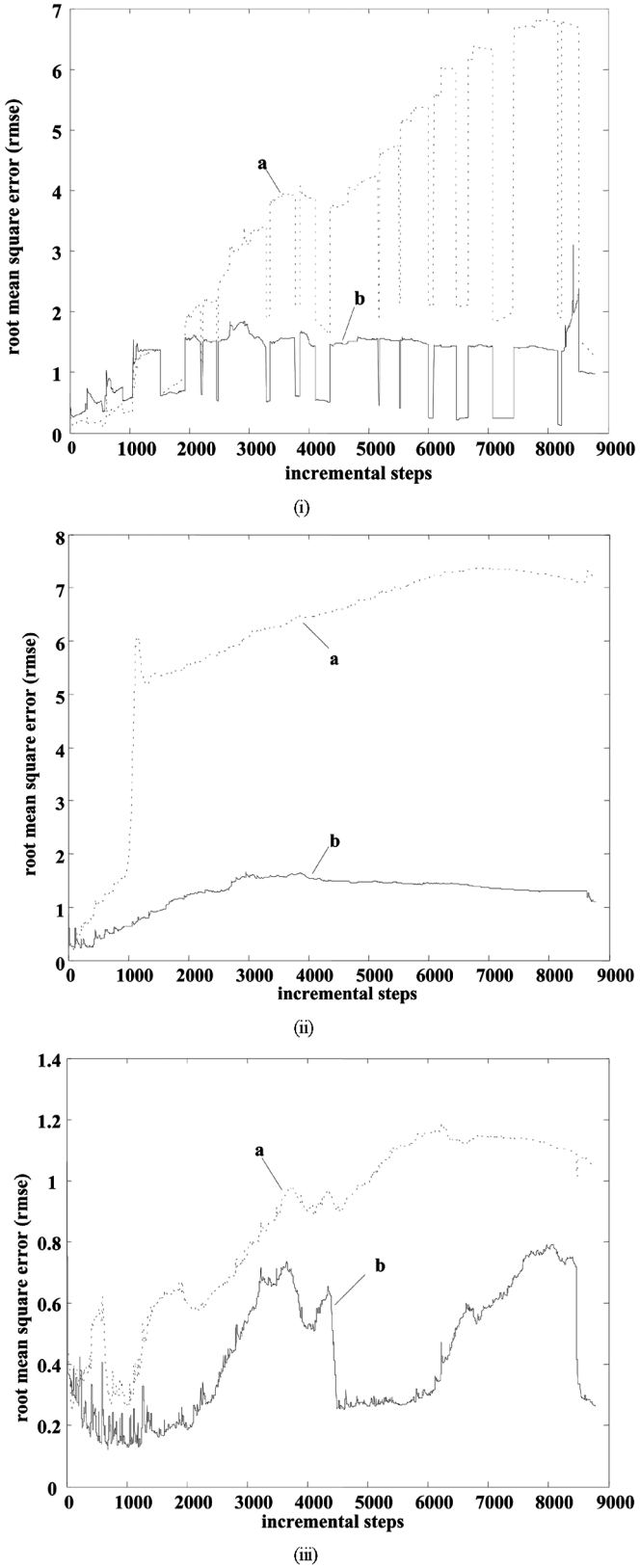


Fig. 9. Comparison of root mean square error for (a) conventional EKF-based SLAM and (b) neuro-fuzzy assisted EKF-based SLAM for case study II ($\sigma_r = 0.01$ m. and $\sigma_b = 3.0$ deg.) with (i) 35, (ii) 135, and (iii) 497 features/landmarks in the environment.

For the neuro-fuzzy assisted EKF based SLAM, the training of the neuro-fuzzy system, for each case study as described before, has been carried out in offline situation on the basis of the data gathered by the robot for a given environment situation. For our experimentation, we implemented the training procedure, for each case study, for the environment containing 135 landmarks. Once the training of the neuro-fuzzy system is completed (on the basis of a given configuration of the landmarks) and the free parameters of the NFS are suitably determined, the trained NFS-based EKF is implemented for robot navigation through the waypoints for several configuration of landmarks as discussed before (i.e., environments with 35, 135 and 497 landmarks). Table I enumerates these parameters, employed for the PSO algorithm employed for training the NFS. Here, the dimensions of each particle, that are deployed to learn the control points of the MFs of the NFS (i.e., $[x_1 \;\; x_2 \;\; \cdots \;\; x_8]$), are all initialized with their positions within the range [-1, 1]. This is done in conformation with the normalization procedure that works in conjunction with the NFS. The prospective weights associated with the layer 3 of the NFS (denoted by the dimensions x_9, x_{10} and x_{11} of the PSO algorithm) are all initialized with their positions within the range [-2, 2]. The prospective gain K associated with the layer 4 of the NFS (denoted by the dimension x_{12} of the PSO algorithm) is initialized with its position within the range $[0, 2]$, as it is assumed that K is a non-negative quantity. Each time, the termination criterion for the PSO algorithm is set for a maximum number of iterations (*maxiter*) of 20. For the case study with initial sensor information $\sigma_r = 2.0$ m, and $\sigma_\theta = 0.1$ deg, the learned parameters of the NFS at the

TABLE I
PSO PARAMETERS EMPLOYED

Sl. No.	Parameter descriptions	Parameter values for case study (*i*)	Parameter value for case study (*ii*)
1	No. of particles (N)	40	40
2	No. of dimensions (D)	12	12
3	Initial inertia weight (W_{initial})	0.9	0.9
4	Slope of inertia weight (W)	2.5e4	2.5e4
5	Initialization range for MFs ($x_1, x_2, \ldots x_8$)	[-1, 1]	[-1, 1]
6	Initialization range for weight factors (x_9, x_{10}, x_{11})	[-2, 2]	[-2, 2]
7	Initialization range for gain (x_{12})	[0, 2]	[0, 2]
8	Maximum permissible velocity for MFs ($v_{1\max}, v_{2\max}, \ldots v_{8\max}$)	0.3	0.1
9	Maximum permissible velocity for weight factors ($v_{9\max}, v_{10\max}, v_{11\max}$)	1.0	0.5
10	Maximum permissible velocity for gain ($v_{12\max}$)	1.0	0.5

$$[x_1 \quad x_2 \quad \cdots \quad x_{12}] = [-0.2008 \quad -0.0626 \quad -0.0626 \quad -0.0626 \quad 0.0820 \quad 0.5961 \quad 0.3224 \quad 0.4002 \quad -0.0086 \quad 1.5801 \quad -0.9729 \quad 0.0011]$$

$$[x_1 \quad x_2 \quad \cdots \quad x_{12}] = [-0.4570 \quad 0.5242 \quad 0.4805 \quad 0.9741 \quad 0.9741 \quad 0.9741 \quad -0.4290 \quad 0.2413 \quad -0.0024 \quad -0.8762 \quad 1.3561 \quad 0.2907].$$

completion of the training procedure are shown in the first equation at the top of the page, and for the case study, with initial sensor information, $\sigma_r = 0.01$ m and $\sigma_\theta = 3.0$ deg, the learned parameters of the NFS are shown in the second equation at the top of the page.

In each case, it can be seen that these learned parameters are satisfying those constraints presented in (37).

VI. Conclusion

The present paper has proposed the development of a neuro-fuzzy assistance for extended Kalman filter based SLAM algorithms, when *a priori* knowledge of the sensor statistics is incorrect. Usually, EKF is known as a good choice for SLAM algorithms when the associated statistical models are well known. However, the performance can become significantly unpredictable and degrading when the knowledge of such statistics is inappropriate. The present scheme proposes to start the system with a wrongly known statistics and then adapt the $\mathbf{R}$ matrix, online, on the basis of a neuro-fuzzy system that attempts to minimize the mismatch between the theoretical and the actual values of the innovation sequence. The free parameters of the neuro-fuzzy system are automatically learned by employing PSO in the training phase. The proposed scheme has been employed for several benchmark environment situations with several wronge knowledge of sensor statistics. While the conventional EKF based SLAM showed unreliable performance with significant degradation in many situations, the proposed neuro-fuzzy assistance can improve this EKF's performance significantly and can provide robust, accurate performance in each sample situation in each case study, described in this work.

Hence, the advantage of the neuro-fuzzy based supervision for the EKF-based SLAM will become really pertinent in those situations where we need to execute the SLAM algorithm in absence of a proper knowledge of the sensor statistics. For solving a SLAM problem, this situation can become extremely tricky as the inaccuracies associated with the estimation of the states, pertaining to the localization of the robot and the positions of the landmarks, can grow very rapidly and there is every possibility that the system may diverge fast. It becomes very difficult or even impossible to control such a system, and, for a leader-follower type configuration of multi-robot systems, this inability of the leading robot to accurately localize and create map will severely hamper the performance or the utility of the following robots. The presence of the proposed neuro-fuzzy supervision can aid the leading robot, in these situations, to obtain a satisfactory solution and this will, in effect, also help immensely in improving the performance of the following robots.

A potential limitation of the proposed system is that, for suitably utilizing the neuro-fuzzy supervision, it is needed to learn its free parameters under offline situations. A superior solution should be able to learn a suitably constructed fuzzy/neuro-fuzzy supervision under online condition, which requires that it should do away with any offline training of its free parameters. At the moment, this problem is viewed as a very difficult problem, specially under the SLAM domain, and the researchers wish to undertake this problem in near future.

The present scheme has been developed under the assumption that an *a priori* knowledge about the $\mathbf{Q}$ matrix is accurate enough. However, there may be such situations where an *a priori* knowledge or estimation of both $\mathbf{Q}$ and $\mathbf{R}$ matrices are inaccurate. The authors intend to undertake this as a further complicated problem for adapting $\mathbf{Q}$ and $\mathbf{R}$ matrices simultaneously, online, as their future study. The choice of the basis on which the $\mathbf{Q}$ matrix should be adapted and how to configure an efficient neuro-fuzzy system for such situations will pose a real challenge.

References

[1] M. W. M. G. Dissanayake, P. Newman, S. Clark, and H. F. Durrant-Whyte, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Tran. Robot. Automat.*, vol. 17, no. 3, pp. 229–241, Jun. 2001.

[2] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proc. 18th Int. Joint Conf. Artif. Intell.*, Acapulco, Mexico, 2003.

[3] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling," in *Proc. IEEE Int. Conf. Robot. Automat.*, Barcelona, Spain, 2005, pp. 2443–2448.

[4] R. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *Int. J. Robot. Res.*, vol. 5, no. 4, pp. 56–68, 1986.

[5] P. Moutarlier and R. Chatila, "Stochastic multisensory data fusion for mobile robot location and environment modeling," in *Proc. 5th Int. Symp. Robot. Research*, Tokyo, Japan, 1989.

[6] A. J. Davison, "Mobile robot navigation using active vision," Ph.D. dissertation, Univ. Oxford, Oxford, U.K., 1998.

[7] T. Bailey, "Mobile robot localization and mapping in extensive outdoor environments," Ph.D. dissertation, Univ. Sydney, Sydney, NSW, Australia, 2002.

[8] A. J. Davison and D. W. Murray, "Simultaneous localization and map-building using active vision," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 7, pp. 865–880, Jul. 2002.

[9] J. Guivant and E. Nebot, "Optimization of the simultaneous localization and map-building algorithm and real-time implementation," *IEEE Trans. Robot. Automat.*, vol. 17, no. 3, pp. 242–257, Jun. 2001.

[10] ——, "Solving computational and memory requirements of feature-based simultaneous localization and mapping algorithms," *IEEE Trans. Robot. Automat.*, vol. 19, no. 4, pp. 749–755, Aug. 2003.

[11] S. B. Williams, P. Newman, G. Dissanayake, and H. Durrant-Whyte, "Autonomous underwater simultaneous localization and map building," in *Proc. IEEE Int. Conf. Robot. Automat.*, San Francisco, CA, 2000, vol. 2, pp. 1792–1798.

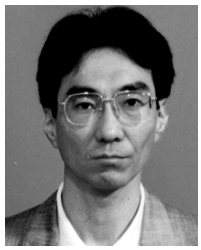
[12] K. S. Chong and L. Kleeman, "Feature-based mapping in real, large scale environments using an ultrasonic array," *Int. J. Robot. Res.*, vol. 18, no. 2, pp. 3–19, Jan. 1999.

[13] M. Bosse, J. Leonard, and S. Teller, J. Leonard, J. D. Tard'os, S. Thrun, and H. Choset, Eds., "Large-scale CML using a network of multiple local maps," in *Workshop Notes of the ICRA Workshop on Concurrent Mapping and Localization for Autonomous Mobile Robots (W4)*, Washington, D.C., 2002.

[14] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots," *Mach. Learn.*, vol. 31, pp. 29–53, 1998.
[15] S. Williams, G. Dissanayake, and H. F. Durrant-Whyte, "Towards terrain-aided navigation for underwater robotics," *Adv. Robot.*, vol. 15, no. 5, pp. 533–550, 2001.
[16] S. Thrun, D. Ĥahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," in *Proc. IEEE Int. Conf. Robot. Automat.*, Taipei, Taiwan, May 2003, pp. 4270–4275.
[17] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós, "The SPmap: A probabilistic framework for simultaneous localization and map building," *IEEE Trans. Robot. Automat.*, vol. 15, no. 5, pp. 948–953, Oct. 1999.
[18] M. A. Paskin, "Thin junction tree filters for simultaneous localization and mapping," in *Proc. 16th Int. Joint Conf. Artif. Intell.*, Acapulco, Mexico, 2003.
[19] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Y. Ng, J. D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, Eds., "Simultaneous mapping and localization with sparse extended information filters," in *Proc. 5th Int. Workshop Algorithmic Foundations Robotics*, Nice, France, 2002.
[20] J. Neira and J. D. Tardós, "Data association in stochastic mapping using the joint compatibility test," *IEEE Trans. Robot. Automat.*, vol. 17, no. 6, pp. 890–897, Dec. 2001.
[21] H. Shatkay and L. Kaelbling, "Learning topological maps with weak local odometric information," in *Proc. 15th Int. Joint Conf. Artif. Intell.*, Nagoya, Japan, Aug. 1997.
[22] A. Araneda, "Statistical inference in mapping and localization for a mobile robot," in *Bayesian Statistics 7*, J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A.F.M. Smith, and M. West, Eds. Oxford, U.K.: Oxford Univ. Press, 2003.
[23] M. Montemerlo and S. Thrun, "Simultaneous localization and mapping with unknown data association using fastSLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, Taipei, Taiwan, 2003, pp. 1985–1991.
[24] W. Hu, T. Downs, G. Wyeth, M. Milford, and D. Prasser, "A modified particle filter for simultaneous robot localization and landmark tracking in an indoor environment," in *Proc. Australian Conf. Robot. Automat.*, Canberra, Australia, 2004.
[25] U. Frese, P. Larsson, and T. Duckett, "A multilevel relaxation algorithm for simultaneous localization and mapping," *IEEE Trans. Robot. Automat.*, vol. 21, no. 2, pp. 196–207, Apr. 2005.
[26] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI National Conf. Artif. Intell.*, Edmonton, AB, Canada, 2002.
[27] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Auton. Robot.*, vol. 4, pp. 333–349, 1997.
[28] R. K. Mehra, "On the identification of variances and adaptive Kalman filtering," *IEEE Trans. Autom. Control*, vol. AC-15, no. 2, pp. 175–184, Apr. 1970.
[29] R. J. Fitzgerald, "Divergence of the Kalman filter," *IEEE Trans. Autom. Control*, vol. AC-16, no. 6, pp. 736–747, Dec. 1971.
[30] N. K. Sinha and A. Tom, "Adaptive state estimation for systems with unknown noise covariances," *Int. J. Syst. Sci.*, vol. 8, no. 4, pp. 377–384, 1977.
[31] P. R. Bellanger, "Estimation of noise covariance matrices for a linear time-varying stochastic process," *Automatica*, vol. 10, pp. 267–275, 1974.
[32] D. P. Dee, S. E. Cohn, A. Dalcher, and M. Ghil, "An efficient algorithm for estimating noise covariances in distributed systems," *IEEE Trans. Autom. Control*, vol. AC-30, no. 11, pp. 1057–1065, Nov. 1985.
[33] R. G. Reynolds, "Robust estimation of covariance matrices," *IEEE Trans. Autom. Control*, vol. 32, no. 9, pp. 1047–1051, Sep. 1990.
[34] H. Morikawa and H. Fujisaki, "System identification of the speech production process based on a state-space representation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-32, no. 4, pp. 252–262, Apr. 1984.
[35] G. Noriega and S. Pasupathy, "Adaptive estimation of noise covariance matrices in real-time preprocessing of geophysical data," *IEEE Trans. Geosci. Remote Sens.*, vol. 35, no. 5, pp. 1146–1159, Sep. 1997.
[36] K. Kobayashi, K. C. Cheok, K. Watanabe, and F. Munekata, "Accurate differential global positioning system via fuzzy logic Kalman filter sensor fusion technique," *IEEE Tran. Ind. Electron.*, vol. 45, no. 3, pp. 510–518, Jun. 1998.
[37] D. Loebis, R. Sutton, J. Chudley, and W. Naeem, "Adaptive tuning of a Kalman filter via fuzzy logic for an intelligent auv navigation system," *Control Eng. Pract.*, vol. 12, pp. 1531–1539, 2004.
[38] Z. Q. Wu and C. J. Harris, "An adaptive neurofuzzy Kalman filter," in *Proc. 5th Int. Conf. Fuzzy Sets Syst.*, Sep. 1996, vol. 2, pp. 1344–1350.
[39] J. Z. Sasiadek, Q. Wang, and M. B. Zeremba, "Fuzzy adaptive Kalman filtering for INS/GPS data fusion," in *Proc. 15th Int. Symp. Intell. Control*, Patras, Greece, Jul. 2000.
[40] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability and convergence in a multidimensional complex space," *IEEE Tran. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.
[41] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization," in *Proc. Congr. Evol. Comput.*, 1999, pp. 1945–1950.
[42] [Online]. Available: http://www.acfr.usyd.edu.au/homepages/academic/tbailey/software/software.html
[43] R. G. Brown and P. Y. C. Hwang, *Introduction to Random Signals and Applied Kalman Filtering*, 3rd ed. New York: Wiley, 1997.

Amitava Chatterjee received the B.E., M.E., and Ph.D. degrees in electrical engineering from Jadavpur University, Kolkata, India, in 1991, 1994, and 2002, respectively.

In 1997, he joined the Department of Electrical Engineering, Jadavpur University, where he is presently serving as a Reader. In 2003, he received the Japanese Government (Monbukagakusho) Scholarship and went to Saga University, Saga, Japan. In early 2004, he was invited as a Teacher-Researcher in the University of Paris XII, Val de Marne, France. From November 2004 to November 2005, he was with the University of Electro-Communications, Tokyo, Japan, on a JSPS Postdoctoral Fellowship for Foreign Researchers. His research interests include intelligent instrumentation and control, intelligent signal and image processing, and robotics. He has published more than 40 technical papers, including 21 international journal papers, in his areas of research interest.



Fumitoshi Matsuno (M'94) was born in Nagoya, Japan, on July 26, 1957. He received the Ph.D. (Dr. Eng.) degree from Osaka University, Osaka, Japan, in 1986.

In 1986, he joined the Department of Control Engineering, Osaka University. He became a Lecturer in 1991 and an Associate Professor in 1992, in the Department of Systems Engineering, Kobe University, Kobe, Japan. In 1996, he joined the Department of Computational Intelligence and Systems Science, Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Tokyo, Japan, as an Associate Professor. Since 2003, he has been a Professor in the Department of Mechanical Engineering and Intelligent Systems, University of Electrocommunications. He is also the Vice President and the Leader of Kawasaki laboratory of International Rescue System Institute. His current research interests lie in robotics, control of distributed parameter system and nonlinear system, rescue support system in fire and disaster, and geographic information system. He has published 84 journal papers, 126 international conference paper which passed peer reviews, 35 survey articles, and nine books.

Dr. Matsuno received the Research Promotion Prize and the Outstanding Paper Award from the Institute of Systems, Control, and Information Engineers of Japan in 1986 and 1993, respectively. He also received the Research Promotion Prize from the Robotic Society of Japan in 1989, the Outstanding Paper Award from the Fire and Disaster Management Agency, the Ministry of Home Affairs of Japan in 1997, the Outstanding Paper Award from the Society of Instrument and Control Engineers, in 2001 and 2006, Takeda Memorial Prize in 2001, among other awards. He is a member of the IEEE, the JSME, the RSJ, the ISCIE, the SICE, and the GISA, among other organizations. He served as a local arrangement chair of the 1999 IEEE International Conference on Systems, Man, and Cybernetics (SMC99), a Program Co-Chair of the 2004 IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR04), an Invited Sessions Co-Chair of 2004 IEEE International Conference on Robotics and Biomimetics (RoBio04), and a Program Chair of SSRR05. He is also a Program Vice Chair of IEEE Conference on Control Applications (CCA2007), a Co-Chair of the IEEE Technical Committee on Safety, Security, and Rescue Robotics, and an Associate Editor of the *International Journal of Advanced Robotics*, the *International Journal of Control, Automation and Systems*. He is on the Confernce Editorial Board of the IEEE CSS.