Modified Neural Network aided EKF based SLAM for improving an accuracy of the feature map

Conference Paper · July 2010

DOI: 10.1109/IJCNN.2010.5596656 · Source: DBLP

3 authors:

Jeong-Gwan Kang
Samsung
25 PUBLICATIONS **329** CITATIONS
SEE PROFILE

Su-Yong An
Electronics and Telecommunications Research Institute
33 PUBLICATIONS **230** CITATIONS
SEE PROFILE

Se-Young Oh
Pohang University of Science and Technology
156 PUBLICATIONS **1,958** CITATIONS
SEE PROFILE

Modified Neural Network Aided EKF based SLAM for Improving an accuracy of the Feature Map

Jeong-Gwan Kang, Su-Yong An, and Se-Young Oh, *Senior Member, IEEE*

Abstract—**In this paper, we address a method for improving accuracy of a Neural Network (NN) aided Extended Kalman Filter (EKF) based SLAM by compensating for an odometric error of a robot. The NN is used for estimating the odometric error and online learning of NN is implemented by augmenting the synaptic weights of the NN as the elements of state vector in the EKF-SLAM process. Due to this trainability, the NN could adapt to systematic error of the robot without any prior knowledge and the proposed NN aided EKF-SLAM is very effective compared to the standard EKF-SLAM method under the colored noise or systematic bias error. Experimental results are presented to validate that our NN aided EKF-SLAM generates more accurate feature map than conventional EKF-SLAM.**

I. Introduction

An autonomous mobile robot must have the ability to navigate in an unknown environment. To navigate in unknown environments, the ability to build an environment map is first required for the mobile robot. However, to build map autonomously in unknown environment is a complex problem in the mobile robot literature, because the robot requires a good pose estimate while at the same time a consistent map is needed to localize the robot. This is a Simultaneous Localization and Mapping (SLAM) problem in mobile robotics.

A number of approaches have been proposed to address the SLAM problem using sonars, laser range finders and vision sensors. Chong and Kleeman [1] achieved impressive results using advanced tracking sonar and an accurate odometry combined with a sub-mapping strategy. Thrun et al. [2] have produced some of the best known demonstrations of robot navigation in real environments using laser range finders. Along the same line, Castellanos [3] also used a laser range finder and a mapping strategy called the SPmap. Vision sensors are attractive for mobile robots because they are information-rich and rarely have restrictions in various applications. Among the recent works reported in the field of SLAM, the system proposed by Davison [4, 5] shows a good advance of the vision based SLAM, which uses an active vision and shows the sensor's attractiveness in real applications.

Much of previous research for SLAM is based on the extended Kalman filter (EKF). Although EKF has been one of the most powerful methods to solve SLAM, it still has problems: the EKF based SLAM is very sensitive to the system and sensor noise model. Therefore, an accurate noise modeling is required. However it is difficult to find accurate motion/sensor models and noise variances in the real robot system. In addition to this, since EKF-SLAM is based on the assumption of white Gaussian noise, if the colored noises or systematic bias errors are applied to the robot, EKF-SLAM cannot guarantee an accurate feature map. For example, if the distance between the left and right wheels or the wheel radius is not precisely known, the time update of EKF-SLAM will lead the estimates to drift. In the absence of sensor measurements, this could cause the estimation errors to grow unbound.

A. Related Works

As we mentioned before, due to the sensitivity of EKF based SLAM to the system and sensor noise model, several methods were proposed to adapt a nominal system/sensor noise to the actual one. Borenstein and Feng [6] divided possible error sources in the robot into two categories including systematic and non-systematic errors and developed a calibration technique called UMB method which calibrates for systematic errors of the differential wheeled mobile robot. Doh et al. [7] proposed a calibration method called PC-method for compensating the systematic error. Antonelli [8] identified 3-parameters odometric model and presented a calibration method based on the lest-squares technique. These odometry calibration methods are off-line method, so the odometry calibration is carried out after traveling a specific path.

Larsen et al. [9] proposed an on-line calibration method for the systematic error using an Augmented Extended Kalman Filter (AEKF) which simultaneously estimates the robot pose and the parameters characterizing the systematic error. Martinelli [10] extended the AEKF based approach which estimates not only systematic errors but also non-systematic errors. However, an accurate feature map is needed to implement the AEKF based approach, because it is based on the EKF localization method.

Manuscript received February 7, 2010. This work was supported by National Strategic R&D Program for Industrial Technology of the Korean Ministry of Knowledge Economy.

J. G. Kang is with the Electrical Engineering Department, Pohang University of Science and Technology, San 31, Hyojadong, Namgu, Pohang, Republic of Korea. (phone: 82-54-279-2904, fax: 82-54-279-5594, e-mail: naroo1@postech.ac.kr).

S. Y. An is with the Electrical Engineering Department, Pohang University of Science and Technology, Pohang, Republic of Korea (e-mail: grasshop@postech.ac.kr).

S. Y. Oh is with the Electrical Engineering Department, Pohang University o f Science and Technology, Pohang, Republic of Korea (e-mail: syoh@postech.ac.kr).

In SLAM application, to adapt the sensor noise variance, Chatterjee and Matsuno [11] have proposed a neuro-fuzzy assisted EKF-SLAM algorithm. When *a prior* knowledge of the sensor statistics is incorrect or invalid, his scheme starts with the wrongly known variance and adapts the sensor noise model on the basis of a neuro-fuzzy system that attempts to minimize the mismatch between the theoretical and the actual sensor noise variances. Although this neuro-fuzzy assisted EKF-SLAM algorithm may be robust for an unknown noise model, they are still fragile to non-zero biased error being based on EKF-SLAM. Begum et al. [12] used the fuzzy logic to model the robot's odometry while representing the uncertainty of the robot pose as a distribution of the sample set. Each sample denotes a hypothesis on the robot pose in the world frame and these hypotheses are processed using an island model genetic algorithm. However, to build a fuzzy logic for representing the uncertainty of the robot pose requires an accurate prior knowledge about the robot and surface type of the environment. Choi et al. [13] proposed a neural network-aided EKF algorithm for SLAM. It uses a neural network to deal with systematic errors in the motion model, thereby capturing the unmodeled dynamics in order to intelligently adapt to different motion models of the robot.

B. Contribution for This Work

The accuracy of feature map from the EKF-SLAM can be improved by using the neural network (NN) to implicitly model the robot's odometric error. In this paper, the NN compensates for the odometric error of the robot and the online learning of NN is implemented in the EKF-SLAM process by augmenting the synaptic weights of NN as the elements of the EKF state vector.

Choi et al. [13] first proposed the NN aided EKF-SLAM to deal with systematic error of the robot. In NN aided EKF-SLAM process, the NN takes the robot pose and control command as its input and outputs the odometric error of the robot and 5-3-3-3 multilayered NN having two hidden layers with sigmoid activation function is used in the simulation. However, it is difficult for the NN aided EKF-SLAM to deal with the odometric error efficiently, because no relationship exists between the robot pose and the odometric error of the robot and the sigmoid activation function is not proper for the case that the robot motion from the odometry is larger than the actual one.

So we propose the modified NN aided EKF (mNNEKF) SLAM by changing the input vector and the activation function of the NN. First, the NN in the mNNEKF-SLAM takes only the robot's control command, $\mathbf{u}(k)=[v_k, \omega_k]^T$, as its input instead of robot pose because the odometric error of the robot is generated from the robot motion. Then the velocity error, $\mathbf{e}(k)=[e_v, e_\omega]^T$ which corresponds to $\mathbf{u}(k)$, is generated from the NN. Second, to deal with the odometric error which is smaller than zero, we changed the activation function of the NN to a hyperbolic tangent function, because its activation range is between -1 and 1. Finally, the structure of the NN is also changed to 2-5-2. This is simpler than the one from NN aided EKF, thus the NN in the mNNEKF-SLAM can be trained more easily via the EKF process.

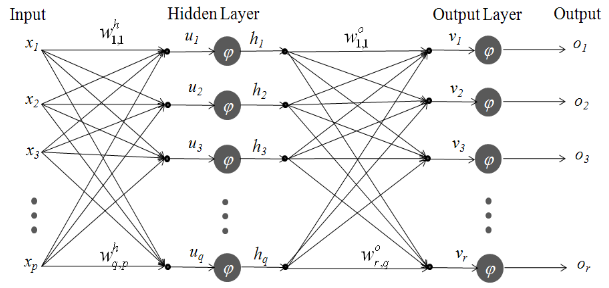


Fig. 1. Multi-layer perceptron neural network architecture.

This paper is organized as follows. Section II presents a short primer on the NN and how to compensate for the odometry error using the NN. Section III addresses the architecture of our mNNEKF-SLAM framework for generating an accurate feature map. In Section IV, the performance of the proposed scheme is evaluated. Finally, conclusions are presented in Section V.

II. Neural Network for Compensating for the Odometric Error

In mobile robotics, NNs have been mainly used to solve navigation and localization problems. In navigation, the NN learns the relationship between the current sensor measurements and the robot's local environment, to plan a collision-free path toward the goal [14, 15]. In localization, the NN learns the location of the feature positions observed from various robot poses [16]. After training, the feature positions are fed as input to the network which then estimates the robot locations. Moreover, the NN outputs the location error by comparing the measurements from the sensor with the predefined global map [17].

A. Neural Network

In our approach, we use a multi-layer perceptron NN (Fig. 1) [18]. This NN is a very flexible nonlinear mapping generator designed to learn the relationship between any input-desired output data pairs where this mapping information is saved in the connection weights between nodes in the NN. An *l*-dimensional input vector can be mapped to an *n*-dimensional output vector by an NN that has *m* hidden nodes. Then, the *k*th element in the output of the NN is computed as:

$$o_k = \varphi\left(\sum_{j=0}^{m-1} w^o_{kj} \cdot \varphi\left(\sum_{i=0}^{l-1} w^h_{ji} x_i\right)\right) \tag{1}$$

$$\varphi(x) = \frac{1-e^{-2x}}{1+e^{-2x}} \tag{2}$$

where $\varphi(x)$ is the hyperbolic tangent activation function to model nonlinearity. In this research, the NN is used to estimate the odometric error of the robot, which is then added to odometry to obtain a more accurate pose estimate.

B. Compensating for the odometric error using the Neural Network

Most error of the robot while traveling is divided into two categories: systematic and non-systematic error. Non-systematic errors are caused by interaction of the robot with unpredictable features of the environment. These non-systematic errors are generated from uneven floor of wheel-slippage due to the slippery floors, over-acceleration, fast turning, external/internal forces and nonpoint wheel contact with the floor [6]. It is almost impossible to predict these non-systematic errors.

Systematic errors are vehicle specific and usually caused by imperfections in the design and mechanical implementation of the robot. In case of differential wheeled robot, several sources which generate systematic errors exist including unequal diameter between left and right wheels, difference between actual average of wheel diameters and nominal diameter, misalignment of wheels and uncertainty about the effective wheelbase. Especially, *unequal wheel diameters* and *the uncertainty about the effective wheel base* are two most notorious systematic error sources [6].

The odometric error of the robot is more affected by the systematic error in most of the indoor environment. Unequal wheel diameter produces a velocity difference between the left and right wheels, making its odometry trajectory lean toward left or right, even when the robot actually moves straight. The uncertainty of the effective wheel base has an effect only when turning.

Let $\mathbf{x}_v(k)=[x_k, y_k, \theta_k]^T$ denote the odometric robot pose in the world frame at time k. Also let $\mathbf{u}(k)=[v_k, \omega_k]^T$ represent the control command. However, because $\mathbf{u}(k)$ is the robot velocity based on odometry, some systematic and random errors are added. So the NN takes $\mathbf{u}(k)$ as its input and outputs the velocity error correction vector, $\mathbf{e}(k)=[e_v, e_\omega]^T$ which, when added to $\mathbf{u}(k)$, yields the compensated robot velocity $\mathbf{u}^*(k)$.

$$\mathbf{u}^*(k)=\mathbf{u}(k)+\mathbf{e}(k)=[v_k+e_v, \omega_k+e_\omega]^T \tag{3}$$

Thus the robot pose corrected by the NN at time k is computed as follows:

$$\begin{bmatrix} x_k^* \\ y_k^* \\ \theta_k^* \end{bmatrix} = \begin{bmatrix} x_{k-1}^* \\ y_{k-1}^* \\ \theta_{k-1}^* \end{bmatrix} + \begin{bmatrix} (v_k+e_v)\cdot dt\cdot\cos(\theta_{k-1}) \\ (v_k+e_v)\cdot dt\cdot\sin(\theta_{k-1}) \\ (\omega_k+e_\omega)\cdot dt \end{bmatrix} \tag{4}$$

III. Modified Neural Network Aided EKF-SLAM

The odometric error from the biased noise can be reduced by compensating for the odometric error of the robot using the NN. The NN is trained online in the EKF-SLAM process by augmenting the synaptic weights of the NN as the elements of the state vector.

In the prediction step, the NN weights in the state vector are converted to generate 2-5-2 structured NN and this NN takes the control command of the robot as its input and outputs the odometric error estimate. Then, the whole state vector of the EKF-SLAM including the synaptic weights of the NN is updated using the Kalman gain and the innovation in the measurement update step. The reduced odometric error in the prediction step makes the feature map of EKF-SLAM more accurate and more accurate feature map guarantees well trained NN for modeling the systematic error of the robot. Finally, the proposed mNNEKF-SLAM can provide an accurate feature map and well trained NN simultaneously after traveling in an unknown environment.

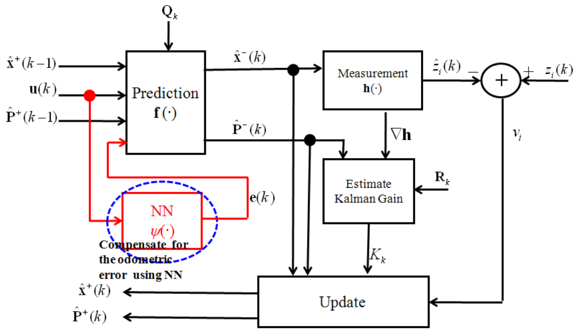


Fig. 2. Overall architecture of mNNEKF-SLAM.

Let $\hat{\mathbf{x}}_w(k)$ be a vector which has the synaptic weights of the NN as its elements, $\hat{\mathbf{x}}_v(k)$ be the estimated robot pose at time k and $\hat{\mathbf{x}}_m(k)$ be the feature position vector which has the positions of the observed features. The state vector in the mNNEKF-SLAM is represented as follows:

$$\begin{aligned}\hat{\mathbf{x}}(k) &= \left[\hat{\mathbf{x}}_v(k), \hat{\mathbf{x}}_w(k), \hat{\mathbf{x}}_m(k)\right]^T \\ &= [\hat{x}_v(k), \hat{y}_v(k), \hat{\theta}_v(k), \hat{w}_{1,1}^h, \cdots, \hat{w}_{q,p}^h, \hat{w}_{1,1}^o, \cdots, \hat{w}_{r,q}^o, \hat{x}_1, \hat{y}_1, \cdots, \hat{x}_n, \hat{y}_n]^T\end{aligned} \tag{5}$$

, where $\hat{w}_{q,p}^h$ is the synaptic weight of the NN between p-th input and q-th hidden node. Similarly $\hat{w}_{r,q}^o$ means the weight between q-th hidden and r-th output node. The covariance matrix $\hat{\mathbf{P}}(k)$ which represents the uncertainty of the robot state vector is defined as follows:

$$\hat{\mathbf{P}}(k) = \begin{bmatrix} \mathbf{P}_{vv} & \mathbf{P}_{vw} & \mathbf{P}_{vm} \\ \mathbf{P}_{vw}^T & \mathbf{P}_{ww} & \mathbf{P}_{wm} \\ \mathbf{P}_{vm}^T & \mathbf{P}_{wm}^T & \mathbf{P}_{mm} \end{bmatrix} \tag{6}$$

The filter recursively updates the mean $\hat{\mathbf{x}}^+(k)$ and covariance $\hat{\mathbf{P}}^+(k)$ by combining the predicted mean $\hat{\mathbf{x}}^-(k)$ and covariance $\hat{\mathbf{P}}^-(k)$ with current noisy measurement $\mathbf{z}(k)$.

The mNNEKF-SLAM algorithm is divided into two steps: prediction and update step (Fig. 2).

1) Prediction Step: In the prediction step, when the robot pose at time k-1 is $\hat{\mathbf{x}}_v^+(k-1)=[\hat{x}_v^+(k-1), \hat{y}_v^+(k-1), \hat{\theta}_v^+(k-1)]^T$ and the control command between time k-1 and k is $\mathbf{u}(k)=[v_k, \omega_k]^T$, the state and covariance of mNNEKF-SLAM at time k are predicted as follows:

$$\mathbf{x}^-(k)=\begin{bmatrix}\hat{\mathbf{x}}_v^-(k)\\ \hat{\mathbf{x}}_w^-(k)\\ \hat{\mathbf{x}}_m^-(k)\end{bmatrix}=\begin{bmatrix}\mathbf{f}\left(\hat{\mathbf{x}}_v^+(k-1),\,\mathbf{u}(k),\mathbf{e}(k)\right)\\ \hat{\mathbf{x}}_w^+(k-1)\\ \hat{\mathbf{x}}_m^+(k-1)\end{bmatrix}\tag{7}$$

, where $\mathbf{e}(k)=[e_v, e_\omega]^T$ is a NN output vector which represents the velocity error corresponding to the $\mathbf{u}(k)$, and $\mathbf{f}(\cdot)$ is a NN aided motion model defined as follows:

$$\begin{aligned}\hat{\mathbf{x}}_v^-(k)&=\mathbf{f}\left(\hat{\mathbf{x}}_v^+(k-1),\,\mathbf{u}(k),\mathbf{e}(k)\right)\\ &=\begin{bmatrix}\mathbf{f}_x\\ \mathbf{f}_y\\ \mathbf{f}_\theta\end{bmatrix}=\begin{bmatrix}\hat{x}_v^+(k-1)\\ \hat{y}_v^+(k-1)\\ \hat{\theta}_v^+(k-1)\end{bmatrix}+\begin{bmatrix}\left(v_k+e_v\right)\cdot dt\cdot\cos\left(\hat{\theta}_v^+(k-1)\right)\\ \left(v_k+e_v\right)\cdot dt\cdot\sin\left(\hat{\theta}_v^+(k-1)\right)\\ \left(\omega_k+e_\omega\right)\cdot dt\end{bmatrix}\end{aligned}\tag{8}$$

$$\begin{aligned}\mathbf{e}(k)&=\begin{bmatrix}e_v\\ e_\omega\end{bmatrix}=\psi\left(\hat{\mathbf{x}}_w^+(k-1),\mathbf{u}(k)\right)\\ &=\varphi\left(\hat{\mathbf{w}}^o\cdot\varphi\left(\hat{\mathbf{w}}^h\cdot\mathbf{u}(k)\right)\right)\\ &=\varphi\left(\begin{bmatrix}\hat{w}_{11}^o & \cdots & \hat{w}_{1q}^o\\ \hat{w}_{21}^o & \cdots & \hat{w}_{2q}^o\end{bmatrix}\cdot\varphi\left(\begin{bmatrix}\hat{w}_{11}^h & \hat{w}_{12}^h\\ \vdots & \vdots\\ \hat{w}_{q1}^h & \hat{w}_{q2}^h\end{bmatrix}\cdot\begin{bmatrix}v_k\\ \omega_k\end{bmatrix}\right)\right)\end{aligned}\tag{9}$$

, where $\varphi(\cdot)$ a hyperbolic tangent activation function and the subscripts p, q and r are the number of nodes in the input, hidden and output layer, respectively. The covariance of the robot state is also predicted as follows:

$$\begin{aligned}\hat{\mathbf{P}}_{k+1}^-&=\nabla\mathbf{f}_\mathbf{x}\hat{\mathbf{P}}_k^+\nabla\mathbf{f}_\mathbf{x}^T+\nabla\mathbf{f}_\mathbf{u}\mathbf{Q}_k\nabla\mathbf{f}_\mathbf{u}^T\\ &=\begin{bmatrix}\nabla_{\mathbf{x}_v}\mathbf{f} & \nabla_{\mathbf{x}_w}\mathbf{f} & 0\\ 0 & \mathbf{I}_{\mathbf{x}_w} & 0\\ 0 & 0 & \mathbf{I}_{\mathbf{x}_m}\end{bmatrix}\cdot\begin{bmatrix}\mathbf{P}_{vv} & \mathbf{P}_{vw} & \mathbf{P}_{vm}\\ \mathbf{P}_{vw}^T & \mathbf{P}_{ww} & \mathbf{P}_{wm}\\ \mathbf{P}_{vm}^T & \mathbf{P}_{wm}^T & \mathbf{P}_{mm}\end{bmatrix}\cdot\begin{bmatrix}\nabla_{\mathbf{x}_v}\mathbf{f} & \nabla_{\mathbf{x}_w}\mathbf{f} & 0\\ 0 & \mathbf{I}_{\mathbf{x}_w} & 0\\ 0 & 0 & \mathbf{I}_{\mathbf{x}_m}\end{bmatrix}^T\\ &\quad+\begin{bmatrix}\nabla_\mathbf{u}\mathbf{f}\\ 0\\ 0\end{bmatrix}\begin{bmatrix}\sigma_v^2 & 0\\ 0 & \sigma_\omega^2\end{bmatrix}\begin{bmatrix}\nabla_\mathbf{u}\mathbf{f}\\ 0\\ 0\end{bmatrix}^T\end{aligned}\tag{10}$$

$$\begin{aligned}\nabla\mathbf{f}_{\mathbf{x}_v}&=\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_v^+(k-1)}\\ &=\begin{bmatrix}1 & 0 & -v_k\cdot dt\cdot\sin\left(\hat{\theta}_v^+(k-1)\right)\\ 0 & 1 & v_k\cdot dt\cdot\cos\left(\hat{\theta}_v^+(k-1)\right)\\ 0 & 0 & 1\end{bmatrix}\end{aligned}\tag{11}$$

$$\nabla\mathbf{f}_\mathbf{u}=\frac{\partial\mathbf{f}}{\partial\mathbf{u}}=\begin{bmatrix}dt\cdot\cos\left(\hat{\theta}_v^+(k-1)\right) & 0\\ dt\cdot\sin\left(\hat{\theta}_v^+(k-1)\right) & 0\\ 0 & dt\end{bmatrix}\tag{12}$$

$$\nabla\mathbf{f}_{\mathbf{x}_w}=\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_w^+(k-1)}=\begin{bmatrix}\frac{\partial\mathbf{f}_x}{\partial w_{1,1}^h} & \cdots & \frac{\partial\mathbf{f}_x}{\partial w_{q,p}^h} & \frac{\partial\mathbf{f}_x}{\partial w_{1,1}^o} & \cdots & \frac{\partial\mathbf{f}_x}{\partial w_{r,q}^o}\\ \frac{\partial\mathbf{f}_y}{\partial w_{1,1}^h} & \cdots & \frac{\partial\mathbf{f}_y}{\partial w_{q,p}^h} & \frac{\partial\mathbf{f}_y}{\partial w_{1,1}^o} & \cdots & \frac{\partial\mathbf{f}_y}{\partial w_{r,q}^o}\\ \frac{\partial\mathbf{f}_x}{\partial w_{1,1}^h} & \cdots & \frac{\partial\mathbf{f}_x}{\partial w_{q,p}^h} & \frac{\partial\mathbf{f}_x}{\partial w_{1,1}^o} & \cdots & \frac{\partial\mathbf{f}_x}{\partial w_{r,q}^o}\end{bmatrix}\tag{13}$$

$$\frac{\partial\mathbf{f}_x}{\partial w_{j,i}^h}=dt\cdot\cos\left(\hat{\theta}_v^+(k-1)\right)\cdot\left(1-e_{v_k}^2\right)\cdot w_{1,j}^o\cdot\left(1-\left(\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\right)^2\right)\cdot x_i\tag{14}$$

$$\frac{\partial\mathbf{f}_x}{\partial w_{k,j}^o}=dt\cdot\cos\left(\hat{\theta}_v^+(k-1)\right)\cdot\left(1-e_{v_k}^2\right)\cdot\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\tag{15}$$

$$\frac{\partial\mathbf{f}_y}{\partial w_{j,i}^h}=dt\cdot\sin\left(\hat{\theta}_v^+(k-1)\right)\cdot\left(1-e_{v_k}^2\right)\cdot w_{1,j}^o\cdot\left(1-\left(\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\right)^2\right)\cdot x_i\tag{16}$$

$$\frac{\partial\mathbf{f}_y}{\partial w_{k,j}^o}=dt\cdot\sin\left(\hat{\theta}_v^+(k-1)\right)\cdot\left(1-e_{v_k}^2\right)\cdot\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\tag{17}$$

$$\frac{\partial\mathbf{f}_\theta}{\partial w_{j,i}^h}=dt\cdot\left(1-e_{\theta_k}^2\right)\cdot w_{1,j}^o\cdot\left(1-\left(\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\right)^2\right)\cdot x_i\tag{18}$$

$$\frac{\partial\mathbf{f}_\theta}{\partial w_{j,i}^h}=dt\cdot\left(1-e_{\theta_k}^2\right)\cdot\varphi\left(\sum_{i=0}^{p-1}w_{j,i}^h x_i\right)\tag{19}$$

, where $\nabla\mathbf{f}_\mathbf{x}$ and $\nabla\mathbf{f}_\mathbf{u}$ are Jacobians of motion model w.r.t the robot state vector and the control command, respectively and $\mathbf{Q}_k$ represents the error covariance matrix of the motion model. In practice, the most common way to increase map accuracy of the EKF-SLAM is to put less confidence in the model and more in the measurements by increasing $\mathbf{Q}_k$. In this paper, the value of $[\sigma_v,\ \sigma_\omega]$ is set to $[0.3m/\sec,\ 3°\cdot\pi/180\,rad/\sec]$ and these are three times larger than actual standard deviation for generating a zero mean Gaussian noise.

2) Update Step: In update step, let $m_i=[\hat{x}_i,\hat{y}_i]^T$ be the position of i th measured feature which already exists in the state vector $\hat{\mathbf{x}}(k)$ and this feature position can be represented as its range (ρ), and bearing (ϕ), based on the predicted robot pose, $\hat{\mathbf{x}}_v^-(k)=[\hat{x}_v^-(k),\hat{y}_v^-(k),\hat{\theta}_v^-(k)]^T$.

$$\begin{aligned}\hat{\mathbf{z}}_i&=\mathbf{h}_i\left(\hat{\mathbf{x}}_v^-(k)\right)=\begin{bmatrix}\hat{\rho}_i\\ \hat{\phi}_i\end{bmatrix}\\ &=\begin{bmatrix}\sqrt{(\hat{x}_i-\hat{x}_v^-(k))^2+(\hat{y}_i-\hat{y}_v^-(k))^2}\\ \tan^{-1}\left(\dfrac{\hat{y}_i-\hat{y}_v^-(k)}{\hat{x}_i-\hat{x}_v^-(k)}\right)-\hat{\theta}_v^-(k)\end{bmatrix}\end{aligned}\tag{20}$$

This feature is measured from the sensor in terms of its range and bearing which are given as $\mathbf{z}_i=[\rho_i,\phi_i]^T$, then we can update mean and covariance of the robot state by following process:

$$\mathbf{S}_k = \nabla\mathbf{h}\hat{\mathbf{P}}^{-}(k)(\nabla\mathbf{h})^T + \mathbf{R}_k \tag{21}$$

$$\mathbf{K}_k = \hat{\mathbf{P}}^{-}(k)(\nabla\mathbf{h})^T(\mathbf{S}_k)^{-1} \tag{22}$$

$$\hat{\mathbf{x}}^{+}(k) = \hat{\mathbf{x}}^{-}(k) + \mathbf{K}_k(z_n - \hat{z}_n) \tag{23}$$

$$\hat{\mathbf{P}}^{+}(k) = (\mathbf{I} - \mathbf{K}_k\nabla\mathbf{h})\hat{\mathbf{P}}^{-}(k) \tag{24}$$

$$\begin{aligned}\nabla\mathbf{h}_i &= \frac{\partial\mathbf{h}_i}{\partial\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}^{-}(k)} \\ &= \begin{bmatrix} \frac{\partial\rho_i}{\partial\hat{x}_v} & \frac{\partial\rho_i}{\partial\hat{y}_v} & \frac{\partial\rho_i}{\partial\hat{\theta}} & \frac{\partial\rho_i}{\partial w^h_{1,1}} & \frac{\partial\rho_i}{\partial w^h_{1,2}} & \cdots & \frac{\partial\rho_i}{\partial w^o_{r,q}} & \frac{\partial\rho_i}{\partial x_1} & \frac{\partial\rho_i}{\partial y_1} & \frac{\partial\rho_i}{\partial x_2} & \cdots \\ \frac{\partial\phi_i}{\partial\hat{x}_v} & \frac{\partial\phi_i}{\partial\hat{y}_v} & \frac{\partial\phi_i}{\partial\hat{\theta}_v} & \frac{\partial\phi_i}{\partial w^h_{1,1}} & \frac{\partial\phi_i}{\partial w^h_{1,2}} & \cdots & \frac{\partial\phi_i}{\partial w^o_{r,q}} & \frac{\partial\phi_i}{\partial x_1} & \frac{\partial\phi_i}{\partial y_1} & \frac{\partial\phi_i}{\partial x_2} & \cdots \end{bmatrix} \\ &= \frac{1}{\rho_i}\begin{bmatrix} -(x_i-\hat{x}_i) & -(y_i-\hat{y}_i) & 0 & 0 & 0 & 0 & \cdots & (y_i-\hat{y}_i) & (x_i-\hat{x}_i) & \cdots \\ \frac{(x_i-\hat{x}_i)}{\rho_i} & \frac{-(y_i-\hat{y}_i)}{\rho_i} & -1 & 0 & 0 & 0 & \cdots & \frac{-(y_i-\hat{y}_i)}{\rho_i} & \frac{(x_i-\hat{x}_i)}{\rho_i} & \cdots \end{bmatrix}\end{aligned} \tag{25}$$

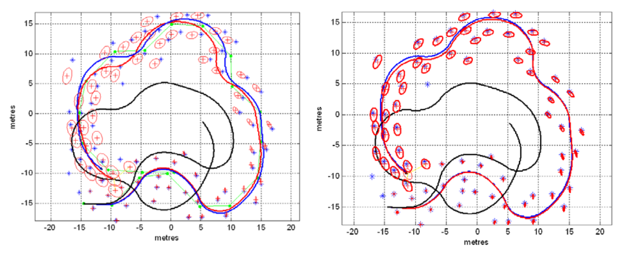
$$\mathbf{R}_k = \begin{bmatrix} \sigma_\rho^2 & 0 \\ 0 & \sigma_\phi^2 \end{bmatrix} \tag{26}$$

, where $\nabla\mathbf{h}_i$ is the Jacobian matrix of the observation model w.r.t the $\hat{\mathbf{x}}^{-}(k)$, $\mathbf{R}_k$ is measurement noise covariance matrix whose elements σ_ρ and σ_ϕ are set to $[0.1m,\ 1°\cdot\pi/180\,rad]$, respectively, and $\mathbf{K}_k$ is a Kalman gain. Then the mean and covariance of the robot state is updated by (23) and (24), respectively.

3) State Augmentation: Let n be the number of features in the state vector $\hat{\mathbf{x}}^{+}(k)$. If the n+1 th feature, $\mathbf{z}_{n+1}$, which does not exist in the state vector, is measured in terms of its range (ρ) and bearing (ϕ) in the robot centered frame, this new feature is added to the state vector through following process. Let $\mathbf{z}_{n+1}=[\rho_{n+1},\ \phi_{n+1}]^T$ be the position of new feature in the robot centered coordinate, then $\mathbf{z}_{n+1}$ is transformed to the world coordinate frame as follows:

$$\begin{aligned}\mathbf{x}_{m_{n+1}} &= g(\hat{\mathbf{x}}_v^{+}(k), \mathbf{z}_{n+1}) \\ &= \begin{bmatrix} \hat{x}_v^{+}(k) + \rho_{n+1}\cos\left(\hat{\theta}_v^{+} + \phi_{n+1}\right) \\ \hat{y}_v^{+}(k) + \rho_{n+1}\sin\left(\hat{\theta}_v^{+} + \phi_{n+1}\right) \end{bmatrix}\end{aligned} \tag{27}$$

Suppose that $\mathbf{x}^*$ is the state vector of the robot which includes a position of newly measured feature,



(a) A feature map and trajectory by the standard EKF-SLAM

(b) A feature map and trajectory by the mNNEKF-SLAM.

Fig. 3. Different feature maps and trajectories from standard EKF and mNNEKF-SLAM with biased noise. Blue solid line: ground truth trajectory, Red solid line: trajectory by EKF/mNNEKF-SLAM, Black solid line: Odometric trajectory, Blue stars: ground truth feature positions, Red crosses and ellipses: feature positions and their uncertainty ellipses by EKF/mNNEKF-SLAM

$$\hat{\mathbf{x}}^*(k) = \begin{bmatrix} \hat{\mathbf{x}}_v^{+}(k) \\ \hat{\mathbf{x}}_{\mathbf{w}}^{+} \\ \hat{\mathbf{x}}_{\mathbf{m}}^{+} \\ g(\hat{\mathbf{x}}^{+}_{k+1,v}, z_{n+1}) \end{bmatrix} \tag{28}$$

and the covariance of the state vector $\hat{\mathbf{x}}^*(k)$ is computed as follows:

$$\begin{aligned}\hat{P}^*(k) &= \nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}}\begin{bmatrix} \hat{\mathbf{P}}^{+}(k) & 0 \\ 0 & \mathbf{R}_k \end{bmatrix}\nabla\mathbf{Y}^T_{\mathbf{x},\mathbf{z}} \\ &= \begin{bmatrix} \mathbf{P}_{vv} & \mathbf{P}_{vw} & \mathbf{P}_{vm} & \mathbf{P}_{vv}^T\mathbf{G}_v^T \\ \mathbf{P}_{vw}^T & \mathbf{P}_{ww} & \mathbf{P}_{wm} & \mathbf{P}_{vw}^T\mathbf{G}_v^T \\ \mathbf{P}_{vm}^T & \mathbf{P}_{wm}^T & \mathbf{P}_{mm} & \mathbf{P}_{vm}^T\mathbf{G}_v^T \\ \mathbf{G}_v\mathbf{P}_{vv} & \mathbf{G}_v\mathbf{P}_{vw} & \mathbf{G}_v\mathbf{P}_{vm} & \mathbf{G}_v\mathbf{P}_{vv}\mathbf{G}_v^T + \mathbf{G}_{\mathbf{z}}\mathbf{R}_k\mathbf{G}_{\mathbf{z}} \end{bmatrix}\end{aligned} \tag{29}$$

$$\nabla\mathbf{Y}_{\mathbf{x},\mathbf{z}} = \begin{bmatrix} \mathbf{I}_v & 0 & 0 & 0 \\ 0 & \mathbf{I}_w & 0 & 0 \\ 0 & 0 & \mathbf{I}_m & 0 \\ \mathbf{G}_v & 0 & 0 & \mathbf{G}_{\mathbf{z}} \end{bmatrix} \tag{30}$$

$$\nabla\mathbf{G}_v = \frac{\partial g}{\partial\mathbf{x}}\Big|_{\mathbf{x}=\hat{\mathbf{x}}_v^{+}(k)}, \nabla\mathbf{G}_{\mathbf{z}} = \frac{\partial g}{\partial\mathbf{z}}\Big|_{\mathbf{z}=\mathbf{z}_{n+1}} \tag{31}$$

IV. Experimental Results

The mNNEKF-SLAM was evaluated both simulated and real indoor environment. For the simulation environment, we used the Matlab simulation developed by Bailey et al. [19]. Since the vehicle model used in Bailey's simulation program was the Ackerman model [20], we modified it to a differential wheel model [21] and also set the maximum detection range of the sensor to 4 m. An update step is carried out after eight consecutive prediction steps and this helps in reducing the computational burden of the SLAM algorithm [19].

The real experiment has been carried out on well-known data sets, named as Freiburg building [22]. In the Freiburg building data set, the size of map is 40×18m and odometry and range measurements were recorded by a Pioneer and a SICK laser scanner. In the middle of the map, a long corridor exists and this is a good for testing the odometric error compensating performance of the EKF and mNNEKF-SLAM. The number of nodes in the input, hidden and output layer in the NN set to 2, 5 and 2, because the NN takes $\mathbf{u}(k)=[v_k, \omega_k]$ as its input and $\mathbf{e}(k)=[e_v, e_\omega]$ is output from the NN.

A. Simulation Result

The simulation was carried out in situation where biased noise was presented to the robot. The simulation environment was a cyclic path spanning a space of approximately 40x40m. The robot starts from the lower left section of Fig. 3 and eventually returns to the start position. In this simulation, the robot traveled more than 108m and stopped just before the loop closing.

To evaluate the performance of mNNEKF-SLAM working under the biased noise, we deliberately added two types of error of the robot's left and right wheel velocities. The first

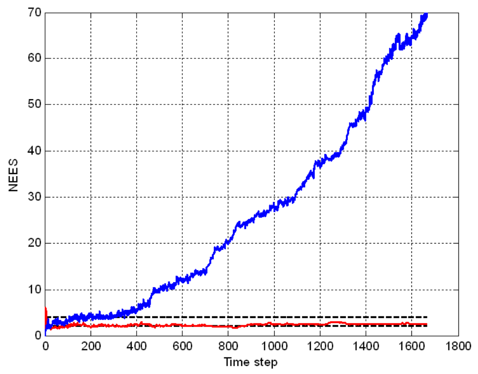


Fig. 4. Average NEES of the robot pose state over 30 Monte Carlo runs. The horizontal dashed line mark the 95% probability concentration region for a 3-dimensional state vector. Blue and red solid lines indicate the average NEES of EKF and mNNEKF-SLAM, respectively.

one was Gaussian noise with zero mean and standard deviation $[\sigma_v,\ \sigma_\omega] = [0.1m/\text{sec}, 1^\circ \cdot \pi/180\,rad/\text{sec}]$. Second one was additional 1% and 2% of the left and right wheel velocities, respectively.

The feature maps were generated using the standard EKF-SLAM and the proposed mNNEKF-SLAM methods (Fig. 3). The RMS error in mNNEKF-SLAM just before loop closing was much smaller than the one in EKF-SLAM (Table I). Because the ground truth for the robot state was available in the simulator, we could test the filter consistency by using the Normalized Estimation Error Squared (*NEES*):

$$\varepsilon_k = \left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right)^T \mathbf{P}_k^{-1} \left(\mathbf{x}_k - \hat{\mathbf{x}}_k\right) \tag{32}$$

Multiple Monte Carlo runs and computing the average *NEES* was carried out for evaluating the consistency of EKF/ mNNEKF. Given N runs, the average *NEES* computed as follows:

$$\hat{\varepsilon}_k = \frac{1}{N}\sum_{i=1}^{N} \varepsilon_{i_k} \tag{32}$$

In this research, 30 Monte Carlo simulations were performed with the two sided 95% *probability concentration region* and it is bounded by the interval [2.19, 3.93]. If $\hat{\varepsilon}_k$ rises significantly higher than the upper bound, the filter is optimistic. If it tends below the lower bound, the filter is conservative. Optimistic means that estimated covariance of the robot is smaller than the actual one [19].

The EKF became optimistic after 400 steps (Fig. 4). This means that the estimated covariance of EKF-SLAM does not contains the ground truth position of the robot and EKF fails to compensate the odometric error accurately. However

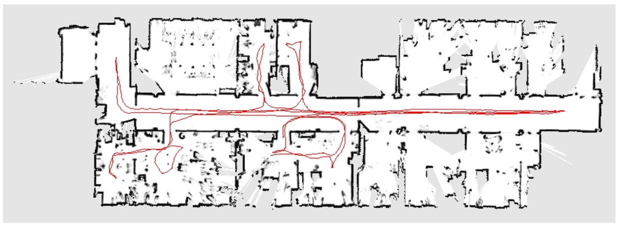


Fig. 5. University of Freiburg, building 079 [22].

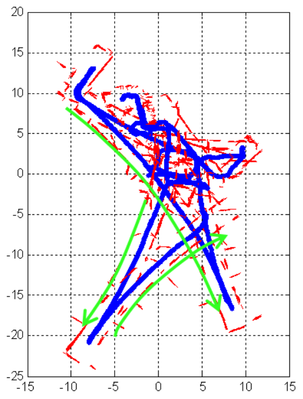
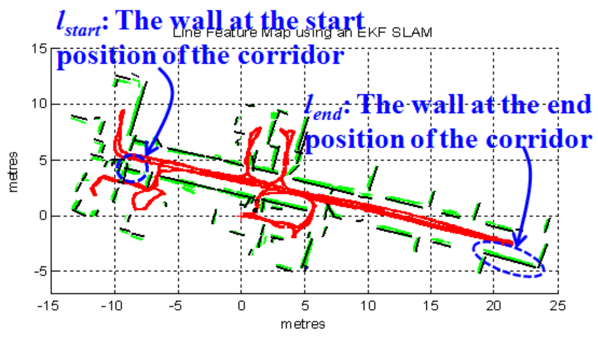
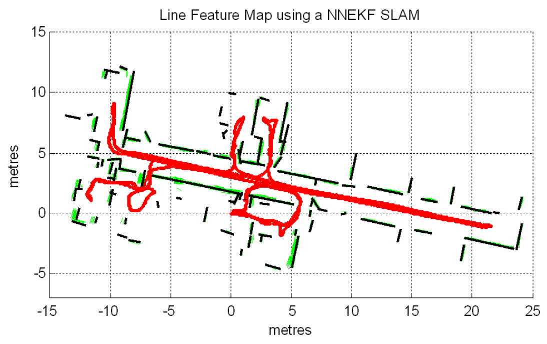


Fig. 6. Robot trajectory and feature map of Freiburg data set based on the odometry. Green solid arrow indicates that the systematic error makes the robot's odometric trajectory lean toward right, even when the robot actually moves straight.



(a) The line based feature map from EKF-SLAM



(b) The line based feature map from mNNEKF-SLAM

Fig. 7. Robot trajectory and feature map of Freiburg data set based on the trajectory by the EKF and mNNEKF-SLAM. Red solid line is the robot trajectory by the SLAM process and black solid line indicates the registered line feature.

mNNEKF did not become optimistic and remained consistent (Fig. 4). These results clearly show that the mNNEKF is superior to standard EKF in compensating for the odometric error, so the proposed mNNEKF is able to generate more

TABLE I
RMS ERROR OF THE EKF/MNNEKF-SLAM

	EKF-SLAM	mNNEKF-SLAM
RMS error (m)	2.223	0.292

accurate feature map than the EKF-SLAM.

B. Real Experiment in the Corridor Environment

Because a long corridor exists in the Freiburg building and the corridor is a rich environment to extract the line segment (Fig. 5), we extracted the line feature from the range measurements data. The systematic error of the robot skews the odometric trajectory to the right, even when the robot actually moves straight (Fig. 6). Then, we generated the line based feature map using EKF and mNNEKF-SLAM (Fig 7). The line feature from the range measurements data, $l_i = [\rho_i,\ \phi_i]^T$, is parameterized by its distance $\rho_i \geq 0$ from the origin and the direction $\phi_i \in (-\pi,\ \pi]$ of the normal passing through the origin, and we adopt the measurement equation of the line feature as in [23].

To check the performance of the odometry error compensation in the real environment, we selected two walls at both ends of the corridor, l_{start} and l_{end} (Fig. 7). If the SLAM algorithm compensates for the odometric error perfectly, the parameters of these two lines should be exactly the same. Let the D_l be the Euclidean distance between l_{start} and l_{end} (Fig. 7a).

$$D_l = \sqrt{\left(\rho_{start} - \rho_{end}\right)^2 + \left(\phi_{start} - \phi_{end}\right)^2} \tag{33}$$

D_l by mNNEKF-SLAM was much smaller than the one by the EKF-SLAM (Table II). This result also shows that the mNNEKF-SLAM can compensate the odometric error more exactly and generate more accurate feature map than the EKF-SLAM in the real environment.

TABLE II
THE LINE PARAMETERS AT BOTH ENDS OF THE CORRIDOR IN FREIBURG DATA SET

	l_{start}	l_{end}	D_l
EKF-SLAM	[1.7359, 1.3063]	[2.3590, 1.2768]	0.6058
mNNEKF-SLAM	[1,7616, 1.3656]	[1.8073, 1.3689]	0.0900

V. CONCLUSION

In this paper, we have proposed the mNNEKF-SLAM for increasing the accuracy of EKF-SLAM. In mNNEKF-SLAM, the NN compensates for the odometric error of the robot and the online learning of NN was implemented in the EKF-SLAM process by augmenting the synaptic weights of NN as the elements of the EKF state vector. So the proposed mNNEKF-SLAM was able to generate a feature map while estimating a proper set of synaptic weights in the NN. For this reason, we could model the systematic error parameters of the robot and reduce the biased motion noise. The experimental results in simulation/real scenarios show that compensating an odometry error by training NN enhances accuracy of the EKF-SLAM and the proposed mNNEKF-SLAM shows a superior accuracy in the feature map than EKF-SLAM.

REFERENCES

[1] K. S. Chong and L. Kleeman, "Feature-based mapping in real large scale environments using a ultrasonic array", *International Journal of Robotics Research*, Vol. 18, no. 2, pp. 3-19, Jan 1999.

[2] S. Thrun, D. Fox, and W. Burgard, "A probabilistic approach to concurrent mapping and localization for mobile robots", *Machine Learning*, vol. 31, 1998.

[3] J. A. Castellanos, "Mobile robot localization and map building: A multisensor fusion approach", PhD thesis, Universidad de Zaragoza, Spain, 1998.

[4] Andrew J. Davison and David W. Murray, "Simultaneous localization and map-building using active vision", *IEEE Trans. Pattern Anal. Mach. Intell.* , Vol. 24, no. 7, pp. 865-880, July 2002.

[5] Andrew J. Davison, "Real-time simultaneous localization and mapping with a single camera", *Proceedings of the 9th IEEE International Conference on Computer Vision*, 2003.

[6] J. Borenstein, and L. Feng, "Measurement and Correction of Systematic Odometry Errors in Mobile Robots," *IEEE. Trans. Robotics and Automation*, vol. 12, no. 6, pp. 869-880, 1996.

[7] N. Doh, H. Choset and W. K. Chung, "Accurate relative localization using odometry," in *Proc. IEEE. Int. Conf. Robotics and Automation,* 2003*,* pp. 1606-1612.

[8] G. Antonelli, S. Chiaverini and G. Fusco, "A systematic calibration method for odometry of mobile robots based on the least-squares technique: Theory and experimental validation," *IEEE. Trans. Robotics and Automation*, vol. 21, no. 5, pp. 994-1004, 2005.

[9] T. D. Larsen, M. Bak, N. A. Andersen and O. Ravn, "Location estimation for autonomously guided vehicle using an augmented Kalman filter to autocalibrate the odometry," in *Proc. FUSION98 Spie Conference*, 1998.

[10] A. Martinelli, N. Tomatis and R. Siegwart, "Simultaneous localization and odometry self calibration for mobile robot," *Autonomous Robots*, vol. 22, pp. 75-85, 2007.

[11] A. Chatterjee and F. Matsuno, "A Neuro-Fuzzy Assisted Extended Kalman Filter-Based Approach for Simultaneous Localization and Mapping (SLAM) Problems," *IEEE. Trans. Fuzzy Systems*, vol. 15, no. 5, pp. 984-997, 2007.

[12] M. Begum, G. K.I. Mann, and R. G. Gosine, "Integrated fuzzy logic and genetic algorithmic approach for simultaneous localization and mapping of mobile robots," *Applied Soft Computing*, vol. 8. pp. 150-165, 2008.

[13] Minyong Choi, Sakthivel. R and Wan Kyun Chung, "Neural Network-Aided Extended Kalman Filter for SLAM Problem," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2007, pp. 1686-1690.

[14] Im KY, Oh SY, Han SJ. (2002) Evolving a Modular Neural Network-Based Behavioral Fusion Using Extended VFF and Environment Classification for Mobile Robot Navigation. IEEE Trans. Evol. Comput. 6(4):413 - 419

[15] Chohra A, Benmehrez C (1998) Neural Navigation Apporach for Intelligent Autonomous Vehicles (IAV) in Partially Structured Environments, Appl. Intell. 8(3):219-233

[16] Asiain JD, Gomez-Allende DM (2001) Landmark Recognition for Autonomous Navigation using Odometric Information and a Network of Perceptrons, Lect. Note in Comput. Sci. 2085: 451-458

[17] Choi WS, Oh SY (2007) Range Sensor-Based Robot Localization Using Neural Network. Int. Conf. Control. Autom. Syst. 1092-1097

[18] Haykin S (2009) Neural Networks, Macmillan, NewYork

[19] T. Bailey, J. Neito, J. Guivant, M. Stevens and E. Nebot, "Consistency of the EKF-SLAM Algorithm," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2006, pp. 3562-3568.

[20] M. A. Sotelo, "Lateral control strategy for autonomous steering of Ackerman-like vehicles," *Robotics and Autonomous Systems,* vol. 24, pp. 223-233, 2003.

[21] G. W. Lucas (2000). A Tutorial and Elementary Trajectory Model for the Differential Steering System of Robot Wheel Actuators [Online]. Available:

[22] M. Batalin, The Robotics Data Set Repository (Radish), http://radish.sourceforge.net.

[23] A. Garulli, et al. "Mobile robot SLAM for line-based environment representation," *IEEE. Conf. Decision and Control and European Control Conference (ICDC)*, pp. 2041-2046, 2005.