2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Neural Network-Aided Extended Kalman Filter for SLAM Problem

Minyong Choi, R. Sakthivel, and Wan Kyun Chung

Abstract— **This paper addresses the problem of Simultaneous Localization and Map Building (SLAM) using a Neural Network aided Extended Kalman Filter (NNEKF) algorithm. Since the EKF is based on the white noise assumption, if there are colored noise or systematic bias error in the system, EKF inevitably diverges. The neural network in this algorithm is used to approximate the uncertainty of the system model due to mismodeling and extreme nonlinearities. Simulation results are presented to illustrate the proposed algorithm NNEKF is very effective compared with the standard EKF algorithm under the practical condition where the mobile robot has bias error in its modeling and environment has strong uncertainties. In this paper, we propose an algorithm which enables a biased control input in vehicle model using neural network.**

I. INTRODUCTION

In robotics, the problem of simultaneous localization and map building is that of estimating both a robot's position and a map of its surrounding environment. In general, it is a complex problem because noise in the estimate of the robot's pose leads to uncertainty in the estimate of the map and vice versa. The Extended Kalman Filter (EKF) has become a standard technique used in simultaneous localization and map building problem. During the past few years significant progress have been made towards the solution for the SLAM problem [3], [4], [6] with the aid of EKF.

EKF assumes a white Gaussian noise in prediction and measurement model. In real situation, however, there are biased systematic error in mobile robot even after appropriate compensation [5]. If there are also some colored noise in control input or drifting errors in dead-reckoning, they could affect the quality of SLAM directly. In addition to these, a priori knowledge of the plant model is required to compute both the prediction of the state estimate and its Jacobian. Often, the plant model is not completely known due to mismodeling, extreme nonlinearities and changes in system parameters. To perform state estimation when such conditions occur, Martinelli et al [10] introduced an augmented Kalman filter which simultaneously estimates the robot configuration and the parameters characterizing the systematic error. In this paper, we propose to implement an EKF whose plant model is augmented by an Neural Network(NN). The NN will capture the unmodeled dynamics by learning on line. The function describes the difference between the true plant and the best model. The processing of imprecise or noisy data by the neural networks is more efficient than classical techniques because neural network is highly tolerant to noises [7]. As a relevant researches in [8], [9], the extended Kalman filter has been used to train the multilayer perception network by treating the weights of a network as the state of the nonlinear dynamic system. In [13] dual extended Kalman filter is presented for removing nonstationary and colored noise from speech. More recently, Zhan and Wan [14] developed NN-aided adaptive unscented Kalman filter for nonlinear state estimation.

Even though the above works studied for state estimation problem, no work have been reported regarding the SLAM problem with the aid of neural network. Motivated by the above considerations, the goal of this paper is to introduce a neural network based extended Kalman filter to SLAM problem to deal with systematic error. To the best of authors Knowledge, this is possibly the first attempt to this problem. The NNEKF is a combination of system identification and standard EKF. In [9], two EKFs are coupled so that a single EKF will be used simultaneously to estimate the state and train the weights of the NN. For simplicity, in this paper we consider a two hidden layer neural network.

This paper is organized as follows. In section II, a neural network aided extended Kalman filter is explained briefly. Section III describes about SLAM problem using NN-aided EKF. In Section IV, simulation results are provided to demonstrate the effectiveness of the proposed algorithm. Finally, Section V contains some concluding remarks.

II. NN-AIDED EXTENDED KALMAN FILTER

A. Neural Network

Neural network is a network of single perceptron. The single perceptron is represented as in Fig. 1. According to inputs x_i and weights a_{ij}, output y_j is changed. The function $f(\cdot)$ is referred to as the squashing function, one of which a sigmoid type function is widely used since its derivative is easily obtained.

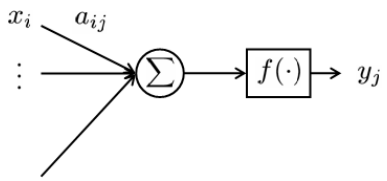


Fig. 1. Single Perceptron

Minyong Choi, R. Sakthivel, and Wan Kyun Chung are with Robotics and Bio-mechatronics Lab., Department of Mechanical Engineering, Pohang University of Science and Technology (POSTECH), Pohang, Korea {minyong,Sakthi,wkchung}@postech.ac.kr

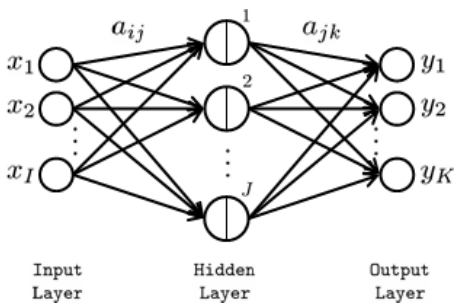


Fig. 2. A Simple Neural Network Structure

$$f(\theta) = \frac{2}{1+e^{-\theta}} + 1 \tag{1}$$

$$f'(\theta) = \frac{1}{2}(1 - f(\theta))^2 \tag{2}$$

A simple neural network which has one hidden layer represented in Fig. 2. Outputs of network are related with inputs, connecting weights, and mapping function like as sigmoid. Each output of NN can be written as the following form

$$y_k = \sum_{j=1}^{J} a_{jk} f\left(\sum_{i=1}^{I} a_{ij} x_i\right) \tag{3}$$

B. NN-Aided Extended Kalman Filter

Lobbia et al. [9] developed an adaptive state estimation technique using NN and they used an EKF to estimate the states by using a dynamic system model, at the same time, using the EKF to train the NN. The NN is used to improve the learning and adaptation capabilities related to variations in the environment where the information is inaccurate, uncertain and incomplete. The plant's model, and observation model are governed by the nonlinear dynamic system

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{f}(\mathbf{x}_k, \mathbf{u}_{k+1}) + \mathbf{w}_{k+1} \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \end{aligned} \tag{4}$$

in which the input vector u_k is the control command, and w_{k+1} and v_k represent process noise and observation noise with zero mean and covariances matrices Q and R respectively. The standard EKF algorithm to estimate the states of the system described in (4) is given by the following equations

$$\begin{aligned} \hat{\mathbf{x}}_{k+1|k} &= \mathbf{f}(\hat{\mathbf{x}}_{k|k}, \mathbf{u}_{k+1}) \\ \mathbf{P}_{k+1|k} &= \nabla_{\mathbf{x}}\mathbf{f} \cdot \mathbf{P}_{k|k} \cdot \nabla_{\mathbf{x}}\mathbf{f}^T + \mathbf{Q}_{k+1} \\ \mathbf{S}_{k+1} &= \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{P}_{k+1|k} \cdot \nabla_{\mathbf{x}}\mathbf{h}^T + \mathbf{R}_{k+1} \\ \mathbf{W}_{k+1} &= \mathbf{P}_{k+1|k} \cdot \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{S}^{-1} \\ \hat{\mathbf{x}}_{k+1|k+1} &= \hat{\mathbf{x}}_{k+1|k} + \mathbf{W}_{k+1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k})) \\ \mathbf{P}_{k+1|k+1} &= \mathbf{P}_{k+1|k} - \mathbf{W}_{k+1}\mathbf{S}_{k+1}\mathbf{W}_{k+1}^T \end{aligned}$$

where $\hat{\mathbf{x}}$ means estimated states, $\mathbf{z}$ represents real measurements, the subscript k is the discrete-time index, $\mathbf{P}$ is estimated error covariance of states, $\mathbf{S}$ means innovation covariance and $\mathbf{W}$ represents the Kalman gain. Roughly, NNEKF is shown in Fig. 3.

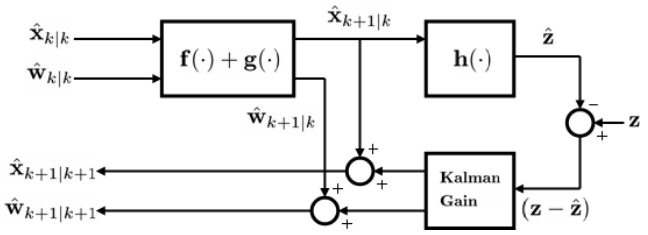


Fig. 3. Neural Network-Aided Extended Kalman Filter

In real situation, true model $f(x_k, u_{k+1})$ is usually unknown quantity. We denote $\mathcal{E}$ as the error function between true model and a priori known mathematical model $\hat{f}(x_k, u_{k+1})$, namely,

$$\mathcal{E} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_{k+1}) - \hat{\mathbf{f}}(\mathbf{x}_k, \mathbf{u}_{k+1}) \tag{5}$$

The NN is used to approximate the error. Obviously if the error $\mathcal{E}$ is small, the state estimate from the NNEKF will be better. We can estimate the error $\mathcal{E}$ using a simple feedforward neural network $\mathbf{g}(\mathbf{x}_k, \mathbf{u}_{k+1}, \mathbf{a}_k)$, $\mathbf{a}$ represent neural network weights. When $\mathcal{E} - \mathbf{g}(\mathbf{x}_k, \mathbf{u}_{k+1}, \mathbf{a}_k) \rightarrow 0$, the error is well approximated and the more accurate model is defined as

$$\begin{bmatrix} \mathbf{x}_{k+1} \\ \mathbf{a}_{k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{f}(\mathbf{x}_k, \mathbf{u}_{k+1}) + \mathbf{g}(\mathbf{x}_k, \mathbf{u}_{k+1}, \mathbf{a}_k) \\ \mathbf{a}_k \end{bmatrix} \tag{6}$$

Now the problem becomes the state estimation based on the above new model and the noisy observation. Now the filtering procedure of NN-aided EKF can be carried out in the similar way of EKF algorithm.

Note: An important advantage of NN-EKF is that the weight is learned during the update, since augmented state vector contains neural networks weight. In this paper, as discussed in [8], [9], [12], two EKFs are coupled into a single EKF which does the double duty by simultaneously estimating the state of the system and training the weights.

III. SLAM USING NNEKF

SLAM is the process of simultaneously building up a map of the environment and using this map to obtain estimates of the vehicle pose. The vehicle is equipped with an onboard sensor to measure relative distance and bearing angle of the vehicle to the environment. As the mobile robot integrate its pose using dead reckoning, it refines its own pose and feature's position from sensor data without absolute information. Since there is noise in odometry information and sensor data, SLAM algorithm requires a quantity to realize the noise like error covariance in EKF.

A. SLAM using standard EKF

We consider 2D SLAM problem, EKF states to solve SLAM are defined as vehicle's pose and features' position;

$$\mathbf{x} = \begin{bmatrix} \mathbf{x_v} \\ \mathbf{x_{f_1}} \\ \vdots \\ \mathbf{x_{f_N}} \end{bmatrix}, \mathbf{x_v} = \begin{bmatrix} x_v \\ y_v \\ \theta_v \end{bmatrix}, \mathbf{x_{f_i}} = \begin{bmatrix} x_{f_i} \\ y_{f_i} \end{bmatrix} \tag{7}$$

where (x, y) is position, and θ is heading of vehicle.

The estimated error covariance is defined as,

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_{vv}} & \mathbf{P_{vf_1}} & \cdots & \mathbf{P_{vf_N}} \\ \mathbf{P_{f_1v}} & \mathbf{P_{f_1f_1}} & \cdots & \mathbf{P_{f_1f_N}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{f_Nv}} & \mathbf{P_{f_Nf_1}} & \cdots & \mathbf{P_{f_Nf_N}} \end{bmatrix} \tag{8}$$

where the diagonal sub-matrices are covariance of vehicle and each feature, and off-diagonal sub-matrices are correlation of them. A detailed SLAM algorithm using standard EKF is presented in [6].

B. SLAM using NNEKF

In this section, the NNEKF is used to study the SLAM problem. A main advantage of the NNEKF is that it is used simultaneously for state estimation (both vehicles position and feature position) and neural network training. If we represent the augmented state vector as $x = [\mathbf{x_v}^T \ \mathbf{x_a}^T \ \mathbf{x_f}^T]^T$ then (6) can be redefined as

$$\begin{bmatrix} \mathbf{x}_{\mathbf{v},k+1} \\ \mathbf{x}_{\mathbf{a},k+1} \\ \mathbf{x}_{\mathbf{f},k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{f_v}(\mathbf{x}_{\mathbf{v},k}, \mathbf{u}_{k+1}) + \mathbf{g}(\mathbf{x}_{\mathbf{v},k}, \mathbf{x}_{\mathbf{a},k}, \mathbf{u}_{k+1}) \\ \mathbf{x}_{\mathbf{a},k} \\ \mathbf{x}_{\mathbf{f},k} \end{bmatrix}$$

where $\mathbf{x_a}$ is neural network's weight.

In general, feature's position in SLAM problem is assumed to be stationary. It is noted that in the prediction model, neural network weights are not changed as like feature's position. The plant Jacobian with respect to states is given by

$$\nabla_{\mathbf{x}}\mathbf{f} = \begin{bmatrix} \nabla_{\mathbf{x_v}}\mathbf{f_v} + \nabla_{\mathbf{x_v}}\mathbf{g} & \nabla_{\mathbf{x_a}}\mathbf{g} & \mathbf{0} \\ \mathbf{0} & \mathbf{I_f} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I_a} \end{bmatrix} \tag{9}$$

In the above Jacobian $\nabla_{\mathbf{x_v}}\mathbf{g}$, and $\nabla_{\mathbf{x_a}}\mathbf{g}$ are depending on neural network structure. The plant Jacobian with respect to control inputs is given by

$$\nabla_{\mathbf{u}}\mathbf{f} = \begin{bmatrix} \nabla_{\mathbf{u}}\mathbf{f} + \nabla_{\mathbf{u}}\mathbf{g} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{10}$$

The augmented state vector estimates the state estimate and weights of the neural networks simultaneously. The NNEKF algorithm for SLAM problem is described.

1) Initialize

$$\mathbf{x_v} = \mathbf{0}, \mathbf{P_{vv}} = \mathbf{0}$$
$$\mathbf{x_a} = \epsilon, \mathbf{P_{aa}} = \lambda\mathbf{I}$$

2) Prediction

$$\begin{bmatrix} \hat{\mathbf{x}}_{\mathbf{v},k+1} \\ \hat{\mathbf{x}}_{\mathbf{a},k+1} \\ \hat{\mathbf{x}}_{\mathbf{f},k+1} \end{bmatrix} = \begin{bmatrix} \mathbf{f_v}(\hat{\mathbf{x}}_{\mathbf{v},k}, \mathbf{u}_{k+1}) + \mathbf{g}(\hat{\mathbf{x}}_{\mathbf{v},k}, \hat{\mathbf{x}}_{\mathbf{a},k}, \mathbf{u}_{k+1}) \\ \hat{\mathbf{x}}_{\mathbf{a},k} \\ \hat{\mathbf{x}}_{\mathbf{f},k} \end{bmatrix}$$

$$\mathbf{P}_{k+1|k} = \nabla_{\mathbf{x}}\mathbf{f} \cdot \mathbf{P}_{k|k} \cdot \nabla_{\mathbf{x}}\mathbf{f}^T + \nabla_{\mathbf{u}}\mathbf{f} \cdot \mathbf{Q} \cdot \nabla_{\mathbf{u}}\mathbf{f}^T$$

3) Data Association

4) Update

$$\nabla_{\mathbf{x}}\mathbf{h} = \begin{bmatrix} \nabla_{\mathbf{x_v}}\mathbf{h} & \mathbf{0_a} & \nabla_{\mathbf{x_f}}\mathbf{h} \end{bmatrix}$$
$$\mathbf{S}_{k+1} = \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{P}_{k+1|k} \cdot \nabla_{\mathbf{x}}\mathbf{h}^T + \mathbf{R}_{k+1}$$
$$\mathbf{W}_{k+1} = \mathbf{P}_{k+1|k} \cdot \nabla_{\mathbf{x}}\mathbf{h} \cdot \mathbf{S}^{-1}$$
$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + \mathbf{W}_{k+1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}_{k+1|k}))$$
$$\mathbf{P}_{k+1|k+1} = \mathbf{P}_{k+1|k} - \mathbf{W}_{k+1}\mathbf{S}_{k+1}\mathbf{W}_{k+1}^T$$

5) New Feature Augmentation

IV. SIMULATION RESULT

In this section, we apply the proposed NNEKF algorithm to SLAM problem and compare it with the standard EKF algorithm. To test the performance of our algorithm, we use the Ackerman vehicle model, in which control inputs are linear velocity and steering angle.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} x_k + \delta t V_{k+1}\cos(\theta_k) \\ y_k + \delta t V_{k+1}\sin(\theta_k) \\ \theta_k + (\delta t V_{k+1}\tan(\phi_{k+1}))/L \end{bmatrix} \tag{11}$$

where V is linear velocity, ϕ is steering angle, $L = 0.5$ m, and $\delta t = 0.025$ s.

Matlab simulation code developed by Bailey et al. [2] for SLAM which is opened on the web-site [15]. We modified their simulation code according to our work and used to verify the algorithm.

The vehicle's unbiased control input has maximum linear velocity as 2.0 m/s, and maximum steering angle as 30°. Noise of control input is assumed zero mean Gaussian with $\sigma_V = 0.1$ m/s, and $\sigma_\phi = 1^\circ$. To simulate biased control input, we add linear velocity by -0.05 m/s, and steering angle by 0.2°. This comes from the following reasoning. Suppose both wheel diameters are 20 cm, the change of 0.5 cm of the wheel diameter can make the 0.05 m/s biases in linear velocity. As the wheel tread is 40 cm and the difference between two wheel diameters is 0.2 cm, nearly 0.2° biases could be occurred in steering angle.

The environment is relatively large about 100 m $\times$ 100 m. Since the sufficient number of features are required to guarantee observability, we used about 100 features and there is one feature in each 10 m $\times$ 10 m area.

In the observation model, range-bearing sensor model are used to measure the feature position and vehicle pose related to its map, whose noise level is 0.1 m in range, 1° in bearing. The sensor range is restricted under 20 m, which can detect all features in front of vehicle.

5-3-3-3 multilayered neural network having two hidden layers with sigmoid function is used in the simulation. Since the input of the neural network is the augmented vector of

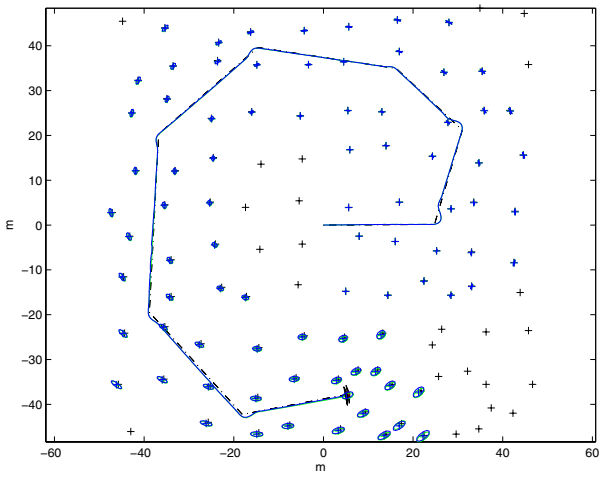


Fig. 4. Simulation Result in Unbiased Case; black dash-dot line: reference trajectory, black (+): true features; blue solid line: estimated trajectory by NNEKF, blue (+): estimated features by NNEKF; blue ellipse: estimated covariance of vehicle and feature by NNEKF; green solid line: estimated trajectory by standard EKF, green (+): estimated features by standard EKF; green ellipse: estimated covariance of vehicle and feature by standard EKF.

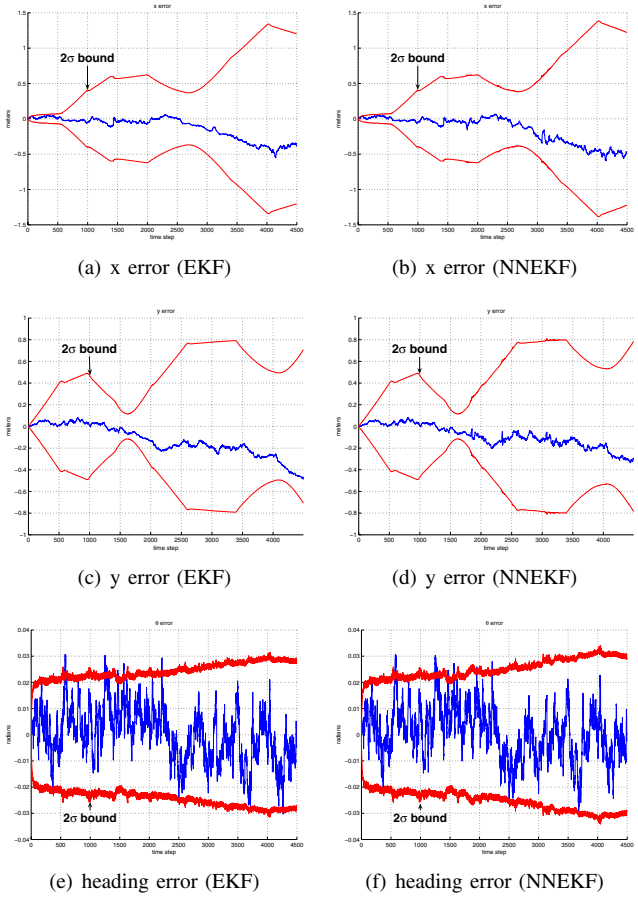
the vehicle pose and control inputs, the number of input neurons is 5. If there is only one hidden layer, an interaction between neurons makes it difficult to learn. So, in practical considerations, two hidden layers are more manageable. The number of neurons of output layer is determined by the dimension of the vehicle pose. Total number of weights are 33, whose initial value is not zero but sufficiently small value.

Data association is also a very important part in SLAM problem [11], and we use a simple nearest neighbor technique since there are sufficient sparseness in features.

In Fig. 4 and Fig. 6, black dash-doted line is the reference trajectory of the vehicle and black crosses are true feature positions. To represent NNEKF case, blue solid line for the estimated trajectory and blue crosses for the estimated feature positions are used. Green solid line for the estimated trajectory and green crosses for the estimated feature positions are used in standard EKF case. Also, blue and green ellipse represent covariance of vehicle and features.

For unbiased control input case, both results are almost the same as shown in Fig. 4, since the noise is white. The vehicle's pose error is plotted in Fig. 5, which shows the consistency of both estimation algorithm.

Fig. 6 shows the biased input case. Initially, both EKF and NNEKF results are almost the same, until neural network's weights are trained. However, as time increases, neural network's weights are trained, significant difference appeared. The error between the true vehicle pose and an estimated vehicle pose is plotted in Fig. 7. In NNEKF, initially the error exceeded 2σ covariance bound, but after some time vehicle's position error is bounded by 2σ as shown in Fig. 7(b) and Fig. 7(d). In EKF, however, the error is not bounded as shown in Fig. 7(a) and Fig. 7(c) after 2500 time step. The result clearly shows that NNEKF is superior to EKF in biased vehicle model condition.



(a) x error (EKF) (b) x error (NNEKF)

(c) y error (EKF) (d) y error (NNEKF)

(e) heading error (EKF) (f) heading error (NNEKF)

Fig. 5. Vehicle Pose Error in Unbiased Model; red lines represent 2σ bound and blue line represents error of each pose

V. CONCLUSIONS

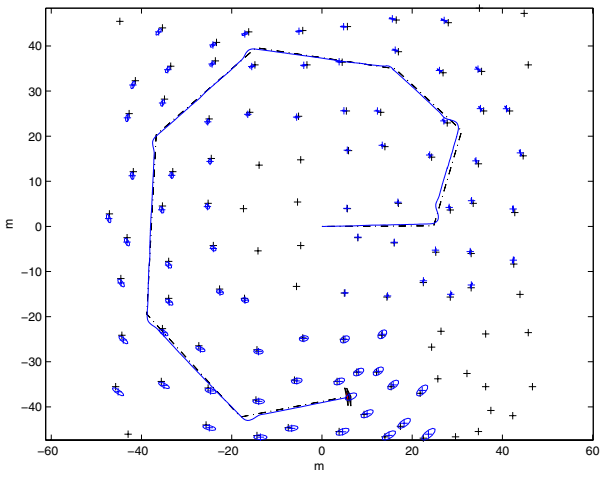
This paper presents a neural network aided extended Kalman filter approach to simultaneous localization and map building problem. As the NN is trained online, NNEKF can capture the unmodeled dynamics and adapt to the changed condition intelligently. The NNEKF was shown to have superior performance with biased vehicle model. Simulation data was used to provide comparison between NNEKF and standard EKF for SLAM problem. The simulation results demonstrate that the proposed NNEKF algorithm is very effective in SLAM problem with biased vehicle model. It is noted, however, that NNEKF and EKF almost have the same performance with unbiased vehicle model.

VI. ACKNOWLEDGMENTS

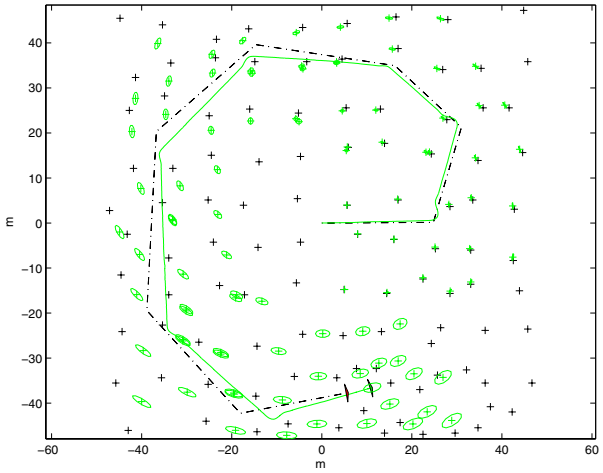
This research was supported in part by the National Research Laboratory (NRL) Program(M1-0302-00-0040-03-J00-00-024-00) of the Ministry of Science and Technology, by a grant(A1100-0602-0026) of the Ministry of Information and Communication, and by a grant(02-PJ3-PG6-EV04-0003) of Ministry of Health and Welfare, Republic of Korea.

REFERENCES

[1] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press Inc., New York, 1995.

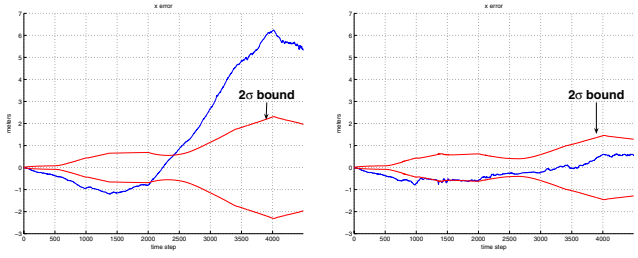


(a) NNEKF

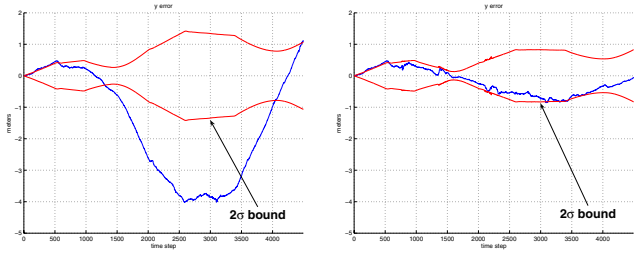


(b) Standard EKF

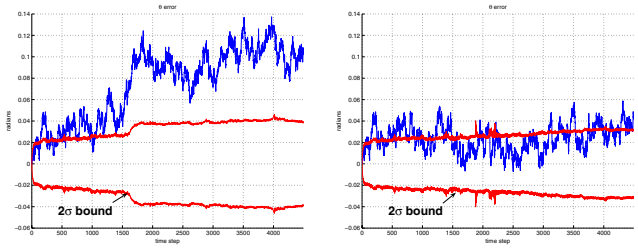
Fig. 6. Simulation Result in Biased Case; black dash-dot line: reference trajectory, black (+): true features; blue solid line: estimated trajectory by NNEKF, blue (+): estimated features by NNEKF; blue ellipse: estimated covariance of vehicle and feature by NNEKF; green solid line: estimated trajectory by standard EKF, green (+): estimated features by standard EKF; green ellipse: estimated covariance of vehicle and feature by standard EKF.



(a) x error (EKF) (b) x error (NNEKF)



(c) y error (EKF) (d) y error (NNEKF)



(e) heading error (EKF) (f) heading error (NNEKF)

Fig. 7. Vehicle Pose Error in Biased Model; red lines represent 2σ bound and blue line represents error of each pose

[2] T. Bailey, J. Neito, J. Guivant, M. Stevens, and E. Nebot, "Consistency of the EKF-SLAM Algorithm," *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 3562-3568.

[3] J. A. Castellanos, J. M. M. Montiel, J. Neira and J.D. Tardos, "The SPmap: A Probabilistic framework for Simultaneous Localization and Map Building," *IEEE Transcations on Robotics and Automation*, Vol. 15, 1999, pp. 948-952.

[4] A. Dissanayake, P. Newman, S. Clark, H. Durrant-Whyte and M. Csorba, "A Solution to the Simultaneous Localization and Map Building Problem," *IEEE Transcations on Robotics and Automation*, Vol. 17, 2001, pp. 229-241.

[5] N. L. Doh, H. Choset, and W. K. Chung, "Accurate Relative Locaization Using Odometry," *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, 2003, pp. 1606-1612.

[6] H. Durrant-Whyle and T. Bailey, "Simultaneous Localization and Mapping: Part I," *IEEE Robotics and Automation Megazine*, Vol. 13, 2006, pp. 99-108.

[7] S. Haykin, Kalman Filtering and Neural Networks, John Wiley and Sons, New York, 2001.

[8] Y. Iiguni, H. Sakai and H. Tokumaru, "A real-time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter," *IEEE Transcations on Signal Processing* , Vol. 40, No. 4, 1992, pp. 959-966.

[9] R. N. Lobbia, S. C. Stubberud, and M. W. Owen, "Adaptive Extended Kalman Filter Using Artificial Neural Networks," *International Journal of Smart Engineering System Design*, Vol. 1, 1998, pp. 207-221.

[10] A. Martinelli, N. Tomatis, A. Tapus, and R. Siegwart, "Simultaneous Localization and Odometry Calibration for Mobile Robot," *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligenct Robots and Systems*, 2003, pp. 1499-1504.

[11] J. Neira and J. D. Tardós, "Data Association in Stochastic Mapping Using the Joint Compatibility Test," *IEEE Transactions on Robotics and Automation*, Vol. 17, No. 6, 2001, pp. 890-897.

[12] D. W. Ruck, S. K. Rogers, M. Kabrisky, P. S. Maybeck, and M. E. Oxley, "Comparative Analysis of Backpropagation and the Extended Kalman Filter for Training Multilayer Perceptrons," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 6, 1992, pp. 686-691.

[13] E. A. Wan and A. T. Nelson, Neural Dual Extended Kalman Filtering: Applications In Speech Enhancement And Monaural Blind Signal Separation, *Proceeding of IEEE workshop on Neural Networks for Signal Processing*, 1997, pp. 466-475.

[14] R. Zhan and J. Wan, "Neural Network-Aided Adaptive Unscented Kalman Filter for Nonlinear State Estimation," *IEEE Signal Processing Letters*, Vol. 13, No. 7, 2006, pp. 445-448.

[15] `http://www.acfr.usyd.edu.au/homepages/academic/tbailey/sofware/index.html`