DOI: 10.1007/s12541-010-0029-9

Enhanced SLAM for a Mobile Robot using Extended Kalman Filter and Neural Networks

Kyung-Sik Choi[1] and **Suk-Gyu Lee**[1,#]

1 Electrical Engineering, Yeungnam University, 214-1, Dae-dong, Gyongsan, Gyongbuk, South Korea, 712-749
# Corresponding Author / E-mail: sglee@ynu.ac.kr, TEL: +82-53-810-3923, FAX: +82-53-810-4767

KEYWORDS: SLAM, Hybrid filter, Neural Networks, EFK, Mobile robot, RBF algorithm

This paper presents a Hybrid filter based Simultaneous Localization and Mapping (SLAM) scheme for a mobile robot to compensate for the Extended Kalman Filter (EKF) based SLAM errors inherently caused by its linearization process. The proposed Hybrid filter consists of a Radial Basis Function (RBF) and EKF which is a milestone for SLAM applications. A mobile robot autonomously explores the environment by interpreting the scene, building an appropriate map, and localizing itself relative to this map. A probabilistic approach has dominated the solution to the SLAM problem, which is a fundamental requirement for mobile robot navigation. The proposed approach, based on a Hybrid filter, has some advantages in handling a robotic system with nonlinear dynamics because of the learning property of the neural networks. The simulation and experimental results show the effectiveness of the proposed algorithm comparing with an EKF based SLAM and Multi Layer Perceptron (MLP) method.

Manuscript received: March 18, 2009 / Accepted: December 8, 2009

NOMENCLATURE

d = center of the basis function
g = the Taylor expansion of function at the motion model
h = nonlinear measurement function
k = time step of a mobile robot navigation
m = vector of a landmark's pose
n = number of input nodes
s = identity of a landmark
u = input vector
vl = velocity of robot's left wheel
vr = velocity of robot's right wheel
v_k = average velocity of robot's wheels
Δv_k = velocity difference between robot's wheels
w = weight of the multi layer perceptron
x = horizontal component of the robot pose
y = vertical component of the robot pose
z = measurement vector
$\hat{z}$ = estimated measurement vector
α = number of the first hidden layer's nodes
β = number of the second hidden layer's nodes
γ = number of output layer's nodes
δt = sampling period
θ = heading angle of the robot
τ = width of the basis function on the radial basis function algorithm
μ = mean
$\bar{\mu}$ = prior mean
$\hat{\mu}$ = mean which results from neural network process
Σ = covariance
$\bar{\Sigma}$ = prior covariance
Π = vector of estimated measurement of a landmark
G = Jacobian of g
H = Jacobian of h
K = Kalman gain
L = width between robot's wheels
M = covariance matrix of the noise in control space
N = the number of hidden layer's nodes on the radial basis function algorithm
Q = covariance of the additional measurement noise
V = Jacobian of g
X = vector of the motion model
Y = vector of the motion model involving map information

1. Introduction

Research efforts on mobile robotics have mainly focused on topics such as obstacle detection, autonomous navigation, path planning, map-building, etc.,[1-3] and many algorithms have been proposed for these purposes. Currently SLAM, which is a relatively new subfield of robotics, is one of the most widely researched major subfields of mobile robotics. In order to solve SLAM problems, statistical approaches, such as Bayesian Filters, have received widespread acceptance. Some of the most popular approaches for SLAM include using a Kalman filter (KF), an extended Kalman filter (EKF) and an unscented Kalman filter (UKF) on which the earliest SLAM was based, and a particle filter.[4-8] As in any EKF based algorithm, the EKF SLAM makes a Gaussian noise assumption for the robot motion and its perception. In addition, the amount of uncertainty in the posterior of the EKF SLAM algorithm must be relatively small; otherwise, the linearization in the EKF tends to introduce intolerable errors. The UKF uses the unscented transform to linearize the motion and measurement models. Especially, the UKF is usually used in order to compensate for the EKF's drawbacks which inherently results from linear approximation of nonlinear functions and the calculation of Jacobian matrices. Differently from the EKF, the main objective of particle filtering is to track a variable of interest as it evolves over time, typically with a non-Gaussian and, potentially, a multi-modal probability density function (PDF). The introduction of particle filters has given researchers the power and flexibility to routinely handle nonlinearities and non-Gaussian distributions. The basis of this method is to construct a sample-based representation of the entire PDF, which is one of the main differences comparing with an EKF based on parameterization.

A neural network (NN), adaptive to the changes of environmental information flowing through the network during the process, can be combined with an EKF to compensate for some of the disadvantages of an EKF SLAM approach, which represents the state uncertainty by its approximate mean and variance, and has biased systematic errors even after appropriate compensation in real situations.[9-13]

Houshangi and Azizi[7] integrated the information from odometry and gyroscope using UKF. To improve the performance of odometry, a fiber optic gyroscope is used to give the orientation information that is more reliable. This method is simple to implement, needless to frequent calibration and applicable to different situations likely EKF. Indeed, they conclude the experiments are performed to examine the approach in estimating the odometry's systematic and nonsystematic errors. The results show that the UKF estimates the robot's position and orientation more accurately than the EKF. According to results of this research, we apply UKF to verify the effectiveness of the Hybrid filer SLAM.

Choi et al[14] approached the SLAM problem with a neural network based on an extended Kalman filter (NNEKF). When the robot is trained online by a NN, the NNEKF can capture the un-modeled dynamics, and adapt to the changed conditions intelligently. According to the research results, the NNEKF SLAM, which uses multi layer perceptrons shows better performance than the EKF SLAM. Stubberud et al[15] developed an adaptive EKF combined with artificial neural networks, with a neuro-observer to learn system uncertainties on-line. The proposed system enhances the overall performance of a control system containing uncertainties in the state-estimator's model. Previous work on the EKF SLAM has shown that an eventual inconsistency of the algorithm is inevitable for large-scale maps. However, these algorithms have a long process time as in MLP, although they have shown enhanced performances.

We propose a Hybrid filter SLAM with RBF which has an advantage on process time under the same conditions. This paper discusses the effectiveness of MLP and a radial basis function (RBF) algorithm to handle nonlinear properties of a mobile robot. In addition, this paper used a biased control input in vehicle model to increase the accuracy, though it needs more time for robot to operate because of the increased complexity in calculation. Several types of Hybrid filter SLAMs (MLP and RBF SLAM with EKF) are proposed to reduce the estimation error comparing with EKF SLAM which is often considered to be the standard SLAM approach. Some related algorithms on SLAM are described in section 2, and the Hybrid SLAM algorithm is presented in section 3. Section 4 shows the simulation results of the SLAM based on EKF, UKF, and two types of Hybrid filter. Section 5 deals with the experiments using EKF SLAM and Hybrid filter SLAM with RBF because other SLAM approaches discussed in chapter 4 shown similar results in simulation. Concluding remarks, discussion and further research are discussed in section 6.

2. Related Algorithms for SLAM

2.1 Neural Networks

Design of artificial NNs is motivated by imitating human brain and thinking activities as a mechanical tool for various purposes. In particular, MLP which was evolved from single layer perceptron with a parallel processing pattern, has been proposed in the early days. However, MLP is turn out to be not suitable to nonlinear information.

On the other hand, RBF has some of advantages on linear time-invariant transfer functions such as low computational complexity, ease of training for finite impulse response models, stability, robustness, and etc.[16]

In this paper, two types of neural networks are considered for SLAM: the Multi Layer Perceptrons Algorithm (MLP) and the Radial Basis Function Algorithm (RBF). The MLP with hidden layers with one or more input and output nodes, is a typical feed-forward neural network model used as a universal approximator. The output signals are generated through the homogeneously nonlinear function after summing signal values for each of the input nodes.[17-20] In this process, signals are multiplied by appropriate weights and added by some bias values. The RBF network uses radial basis functions as the activation functions for function approximation, control, etc. RBF networks typically have three layers, namely, an input layer, a hidden layer with a nonlinear RBF activation function, and a linear output layer. Network training is divided into two stages: first, the weights from the input to the hidden layer are determined; then, the weights from the hidden layer to the output layer are determined. The results can be used to simulate the nonlinear relationship between the sensors' measurements with the errors, and the ideal output values by using the least squares method.[10,17]

2.2 EKF SLAM

A solution to the SLAM problem using EKF, with many interesting theoretical advantages, is extensively described in the research literature. This is despite the recently reported inconsistency of its estimation because it is a heuristic for the nonlinear filtering problem. Associated with the EKF is the Gaussian noise assumption, which significantly impairs the EKF SLAM's ability to deal with uncertainty. With a greater amount of uncertainty in the posterior, the linearization in the EKF fails. An EKF based on a Bayes filter has two steps, prediction and update, for SLAM using the measured sensor data of a mobile robot.[21,22]

3. A Hybrid Filter SLAM Algorithm

A new Hybrid filter SLAM with EKF is proposed here, augmented by an artificial neural network (ANN) acting as an observer to learn the system uncertainty on-line. An adaptive state estimation technique using an EKF and a NN has been developed. In this research, the mobile robot with encoder values (vr_k, vl_k) is learned using NN in the update-step. The mean μ_k, which is derived from environmental information values (x_k, y_k, θ_k) using the NN algorithm, is entered to the prediction-step, as shown in Fig. 1.

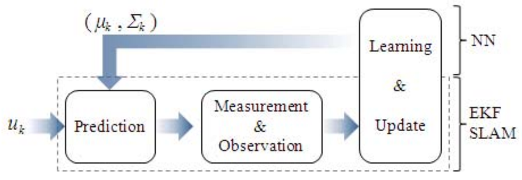


Fig. 1 The architecture of the Hybrid filter SLAM

In this paper, we combine two algorithms on observation step. Basic inputs are mean, covariance which are calculated by prior input, u_{k-1}, and present input, u_k. The robot calculates the prior mean and covariance in a prediction step, and then, in an observation step, it calculates a Kalman gain, present mean and covariance and defined features. At last, the robot is applied to the NN algorithm and update as shown in Fig. 2.

NN algorithm is very important, as the kernel of the Hybrid filter is the complementation of errors onto stochastic EKF SLAM processing through the training process. RBF networks can operate as a fast and accurate means of approximating a nonlinear mapping based on observed data. In the EKF SLAM, we can supply the measurement data through some types of sensors, so it is very useful to train on RBF. Training process, calculating the weight through the entered data from sensors and applying to the robot's estimated pose, are useful to improve the accuracy by reducing the robot pose's errors.

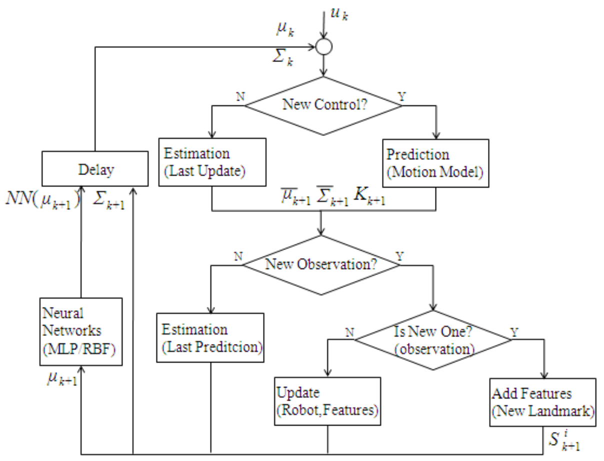

u_k
μ_k
Σ_k
New Control?
N
Y
Delay
Estimation (Last Update)
Prediction (Motion Model)
$NN(\mu_{k+1})$
Σ_{k+1}
$\overline{\mu}_{k+1}$ $\overline{\Sigma}_{k+1}$ K_{k+1}
New Observation?
Neural Networks (MLP/RBF)
Estimation (Last Preditcion)
Is New One? (observation)
μ_{k+1}
Update (Robot,Features)
Add Features (New Landmark)
S^i_{k+1}

Fig. 2 Flowchart of the Hybrid filter SLAM algorithm

NN, especially, applied on observation step is very useful to reduce calculation time. If the robot stays on a prediction step, it requires all of the input data which may include unnecessary information. However, the proposed algorithm needs to learn about necessary information through observation step in the proposed algorithm. Through this way, the calculation time is reduced.

3.1 A Motion Model

The Hybrid filter SLAM algorithm is described using a robot's pose and features, such as the location of landmarks. The estimated error covariance is defined as in Eq. (1), where the diagonal sub-matrices are the covariance of the vehicle and its features. The off-diagonal-matrices are their correlation

$$\Sigma = \begin{pmatrix} \Sigma_{vv} & \Sigma_{vL1} & \cdots & \Sigma_{vLn} \\ \Sigma_{L1v} & \Sigma_{L1L1} & \cdots & \Sigma_{L1Ln} \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma_{Lnv} & \Sigma_{LnL1} & \cdots & \Sigma_{LnLn} \end{pmatrix} \tag{1}$$

For the SLAM, the basic motion model of the mobile robot needs to be presented. A configuration of the robot with a state equation $X = (x\ y\ \theta)^T$ has the form of eq. (2) since it is assumed that the robot is equipped with encoders and exteroceptive sensors.

$$X_{k+1} = \begin{pmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \end{pmatrix} = \begin{pmatrix} x_k + v_{k+1}\delta t\cos(\theta_{k+1}) \\ y_k + v_{k+1}\delta t\sin(\theta_{k+1}) \\ \theta_k + \tan^{-1}\dfrac{\Delta v_{k+1}\delta t}{L} \end{pmatrix} \tag{2}$$

$$u_{k+1} = \begin{pmatrix} vr_{k+1} \\ vl_{k+1} \end{pmatrix} + \mathrm{N}(0, M_{k+1}) \tag{3}$$

where vl_k and vr_k are the velocity of the left and right wheel, and Δv_k and v_k mean the velocity difference between both wheels and average velocity between the robot's wheels, measured by the signals of the motor encoders, L is the width between the robot's wheels, and δt is the sampling period. Finally, M_k describes the covariance matrix of the noise in control space.

The state equation for landmarks, combined with the robot position, is denoted by the vector Y_k, where c in Eq. (4) is the number of landmarks.

$$Y_k = \begin{pmatrix} X_k \\ m \end{pmatrix} = (x_k y_k \theta_k \quad m^1_{k,x} m^1_{k,y} s^1_k \quad m^2_{k,x} m^2_{k,y} s^2_k \quad \cdots \quad m^c_{k,x} m^c_{k,y} s^c_k)^T \tag{4}$$

The state transition probability of a Hybrid filter-SLAM has the form of eq. (5) under the linearity assumption where g represents the nonlinear functions, ε_k is the process noise, and u_k is the combined two elements (vr_k, vl_k) which are velocities of the two wheels of the mobile robot.

$$X_{k+1} = g(X_k, u_k) + \varepsilon_k \tag{5}$$

For the Taylor expansion of function g, its partial derivative is used with respect to x_k, as shown in Eq. (5).

$$g'(X_k, u_{k+1}) = \frac{\partial g(X_k, u_{k+1})}{\partial X_k} \tag{6}$$

g is approximated at μ_k and u_{k+1}. The linear extrapolation is achieved by using the gradient of g at μ_k and u_{k+1} as shown in eq. (7).

$$\begin{aligned} g(X_k, u_{k+1}) &\approx g(\mu_k, u_{k+1}) + g'(\mu_k, u_{k+1})(X_k - \mu_k) \\ &= g(\mu_k, u_{k+1}) + G_{k+1}(X_k - \mu_k) \end{aligned} \tag{7}$$

where G_{k+1}, a Jacobian, is a matrix with dimension $n \times n$, where n denotes the state dimension. It has a different value at each μ_k and u_{k+1}.

$$G_{k+1} = \frac{\partial g(\mu_k, u_{k+1})}{\partial X_k} = \begin{pmatrix} 1 & 0 & -v_{k+1}\delta t \sin(\theta_{k+1}) \\ 0 & 1 & v_{k+1}\delta t \cos(\theta_{k+1}) \\ 0 & 0 & 1 \end{pmatrix} \tag{8}$$

The transformation from the control space to the state space V_{k+1} needs a linear approximation, as shown in eq. (9), where the Jacobian is the derivative of the motion function g with respect to the motion elements, evaluated at μ_k and u_{k+1}.

$$V_{k+1} = \frac{\partial g(\mu_k, u_{k+1})}{\partial u_{k+1}} = \begin{pmatrix} \frac{\delta t \cos(\theta_{k+1})}{2} & \frac{\delta t \cos(\theta_{k+1})}{2} \\ \frac{\delta t \sin(\theta_{k+1})}{2} & \frac{\delta t \sin(\theta_{k+1})}{2} \\ \frac{L\delta t}{L^2 + (\Delta v_{k+1}\delta t)^2} & -\frac{L\delta t}{L^2 + (\Delta v_{k+1}\delta t)^2} \end{pmatrix} \tag{9}$$

3.2 The Prediction Step

The NN algorithm is applied to the observation step of the EKF SLAM to lessen the error of the mobile robot's pose. The prior mean $\bar{\mu}_{k+1}$ and covariance $\bar{\Sigma}_{k+1}$ have the form of

$$\bar{\mu}_{k+1} = g(\mu_k, u_{k+1}) = \mu_k + F_X^T \begin{pmatrix} v_{k+1}\delta t \cos(\theta_{k+1}) \\ v_{k+1}\delta t \sin(\theta_{k+1}) \\ \tan^{-1} \frac{\Delta v_{k+1}\delta t}{L} \end{pmatrix} \tag{10}$$

The motion model requires the motion noise to be mapped into the state space. The Jacobian needed for the approximation, denoted as V_{k+1}, is the derivative of the motion function g, with respect to the motion parameters, evaluated at μ_k and u_{k+1}. Here, F_X is a matrix that maps the 3-dimensionl state vector into a vector of dimension $3N$+3. Also, the Jacobian G_{k+1} needs to extend the dimensions with a matrix F_X.

$$F_X = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \underbrace{0 \quad \cdots \quad 0}_{3N\ columns} \end{pmatrix} \tag{11}$$

$$V_{k+1} = \frac{\partial g(\mu_k, u_{k+1})}{\partial u_{k+1}} \tag{12}$$

$$G_{k+1} = I + F_X^T g_{k+1} F_X \tag{13}$$

$$g_{k+1} = \begin{pmatrix} 0 & 0 & -v_{k+1}\delta t \sin(\theta_{k+1}) \\ 0 & 0 & v_{k+1}\delta t \cos(\theta_{k+1}) \\ 0 & 0 & 0 \end{pmatrix} \tag{14}$$

Eq. (14) describes a low-dimensional Jacobian that characterizes the change of the robot's position. An identity matrix has a dimension of $2 \times (3N+3)$. In addition, the prior covariance $\bar{\Sigma}_{k+1}$ has the form of Eq. (15).

$$\bar{\Sigma}_{k+1} = G_{x,k+1} \Sigma_k G_{x,k+1}^T + V_{k+1} M_{k+1} V_{k+1}^T \tag{15}$$

3.3 The Measurement Step

As described in eq. (16), the measurement probability, z_{k+1} consists of the nonlinear measurement function h and the observation noise δ_k.

$$\begin{aligned} z_{k+1} &= h(Y_{k+1}) + \delta_{k+1} = h(Y_{k+1}, s^i_{k+1}, m^i) + \mathrm{N}(0, Q_{k+1}) \\ &= \begin{pmatrix} r^i_{k+1} \\ \phi^i_{k+1} \\ s^i_{k+1} \end{pmatrix} = \begin{pmatrix} \sqrt{(m^i_{k+1,x} - x_{k+1})^2 + (m^i_{k+1,y} - y_{k+1})^2} \\ \tan^{-1}\left(\frac{m^i_{k+1,y} - y_{k+1}}{m^i_{k+1,x} - x_{k+1}} \right) - \theta_{k+1} \\ s^i_{k+1} \end{pmatrix} \\ &+ \mathrm{N}(0, Q_{k+1}) \end{aligned} \tag{16}$$

It is assumed that s^i_{k+1} is the identity of the *i-th* landmark m^i in the measurement vector. s^i_{k+1} is a set of correspondence variables which have the true identity of an observed feature. The measurement function h is an expansion of g, and the Taylor expansion is developed around $\bar{\mu}_{k+1}$.

$$m^i = (m^i_x \quad m^i_y)^T \tag{17}$$

3.4 The Observation Step

To derive the Kalman gain K_k, it needs to confirm the measurement noise covariances and the measurement model for the feature-based maps. The covariance Q_{k+1} of the additional measurement noise is described as eq. (18).

$$Q_{k+1} = \begin{pmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\phi^2 & 0 \\ 0 & 0 & \sigma_s^2 \end{pmatrix} \tag{18}$$

$$h'(Y_{k+1}) = \frac{\partial h(Y_{k+1})}{\partial Y_{k+1}} \tag{19}$$

$$\begin{aligned} & h(Y_{k+1}, s_{k+1}^i, m^i) \\ & \approx h(\bar{\mu}_{k+1}, s_{k+1}^i, m^i) + h'(\bar{\mu}_{k+1})(Y_{k+1} - \bar{\mu}_{k+1}) \\ & = h(\bar{\mu}_{k+1}, s_{k+1}^i, m^i) + H_{k+1}^i(Y_{k+1} - \bar{\mu}_{k+1}) \end{aligned} \tag{20}$$

The Jacobian H_{k+1}^i of the measurement function *h* which has a low-dimensional matrix h_{k+1}^i depends on two elements of the state vector. The robot's position and the location of the *i-th* landmark are calculated at the prior mean $\bar{\mu}_{k+1}$ with the scalar q as shown in eq. (21).

$$q = (m_{k+1,x}^i - x_{k+1})^2 + (m_{k+1,y}^i - y_{k+1})^2 \tag{21}$$

After the robot finds the landmark, the dimension of the matrix F_x is augmented to $6 \times (3N+3)$. It maps the low-dimensional matrix h_{k+1}^i into a matrix of dimension $3 \times (3N+3)$.

$$F_X^i = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \underbrace{0 & \cdots & 0}_{3i-3\ columns} & 0 & 0 & 1 & \underbrace{0 & \cdots & 0}_{3N-3i} \end{pmatrix} \tag{22}$$

Then, the initial estimation $\hat{\Pi}_\mu^i$ is derived, and the estimated measurement $\hat{z}_{k+1}^i$ of the *i-th* landmark m^i. The expected position can be derived from the expected robot position and the measurement elements of the *i-th* landmark.

$$\hat{\Pi}_\mu^i = \begin{pmatrix} m_{k+1,x}^i \\ m_{k+1,y}^i \\ s_{k+1}^i \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{k+1,x} \\ \bar{\mu}_{k+1,y} \\ s_{k+1}^i \end{pmatrix} + \begin{pmatrix} r_{k+1}^i \cos(\phi_{k+1}^i + \bar{\mu}_{k+1,\theta}) \\ r_{k+1}^i \sin(\phi_{k+1}^i + \bar{\mu}_{k+1,\theta}) \\ 0 \end{pmatrix} \tag{23}$$

$$\hat{z}_{k+1}^i = \begin{pmatrix} \sqrt{(m_{k+1,x}^i - x_{k+1})^2 + (m_{k+1,y}^i - y_{k+1})^2} \\ \tan^{-1}\left(\dfrac{m_{k+1,y}^i - y_{k+1}}{m_{k+1,x}^i - x_{k+1}}\right) - \theta_{k+1} \\ s_{k+1}^i \end{pmatrix} \tag{24}$$

$$\begin{aligned} H_{k+1}^i = h_{k+1}^i F_X &= \frac{\partial h(\bar{\mu}_{k+1}, s_{k+1}^i, m^i)}{\partial Y_k} {}^i F_X \\ &= \begin{pmatrix} \dfrac{\bar{\mu}_{k+1,x} - m_{k+1,x}^i}{\sqrt{q}} & \dfrac{\bar{\mu}_{k+1,y} - m_{k+1,y}^i}{\sqrt{q}} & 0 \\ \dfrac{m_{k+1,y}^i - \bar{\mu}_{k+1,y}}{q} & \dfrac{\bar{\mu}_{k+1,x} - m_{k+1,x}^i}{q} & -1 \\ 0 & 0 & 0 \end{pmatrix} \\ &\quad \begin{matrix} \dfrac{m_{k+1,x}^i - \bar{\mu}_{k+1,x}}{\sqrt{q}} & \dfrac{m_{k+1,y}^i - \bar{\mu}_{k+1,y}}{\sqrt{q}} & 0 \\ \dfrac{\bar{\mu}_{k+1,y} - m_{k+1,y}^i}{q} & \dfrac{m_{k+1,x}^i - \bar{\mu}_{k+1,x}}{q} & 0 \\ 0 & 0 & 1 \end{matrix} \Bigg) F_X \end{aligned} \tag{25}$$

Using the covariance Q_{k+1} and a diagonal matrix with elements of z_{k+1}^i, the Kalman gain has the form of Eq. (26).

$$K_{k+1} = \bar{\Sigma}_{k+1} H_{x,k+1}^T (H_x \bar{\Sigma}_{k+1} H_{x,k+1}^T + Q_{k+1})^{-1} \tag{26}$$

3.5 The Update Step

In this step, two algorithms with EKF are considered to complete SLAM of the mobile robot. One is the combination of MLP and EKF, and the other is the combination of RBF and EKF. Both cases are involved with train through input data and measurement values. In the training process, weights are decided based on the relation of input data and each hidden layers; from among robot's poses and heading angle to relevance among each elements of hidden layers. NN needs higher weight to objective value on the higher relations between poses and heading angle with comparing to measurement.

To apply a MLP, the mean values for each element are divided, and substituted by inputs of the MLP algorithm for each mean value. This research utilizes the MLP with two hidden layers, so the process equation is derived as Eq. (27). Under the assumption that this process does not have any bias, the *n*, α, β, and γ describe the number of input nodes, the first hidden layer's nodes, the second hidden layer's nodes and output layer's nodes with A, B and C, the number of nodes, respectively.

$$\begin{aligned} \hat{\mu}_k^n &= \zeta\left(\sum_{\gamma=0}^{C-1} w_k^{\beta\gamma} \varphi_k^\beta\right) \\ &= \zeta\left(\sum_{\gamma=0}^{C-1} w_k^{\beta\gamma} \zeta\left(\sum_{\beta=0}^{B-1} w_k^{\alpha\beta} \varphi_k^\alpha\right)\right) \\ &= \zeta\left(\sum_{\gamma=0}^{C-1} w_k^{\beta\gamma} \zeta\left(\sum_{\beta=0}^{B-1} w_k^{\alpha\beta} \zeta\left(\sum_{\alpha=0}^{A-1} w_k^{n\alpha} \bar{\mu}_k^n\right)\right)\right) \\ & (0 \le \alpha \le A-1,\ 0 \le \beta \le B-1,\ 0 \le \gamma \le C-1) \end{aligned} \tag{27}$$

When applying the other case for RBF, it is the same to substitute inputs. The RBF algorithm generally consists of two weight layers; one hidden layer and the output layer. In addition, the second weight, ω_0, equals zero because the output offset is zero. Therefore, new estimated mean, $\hat{\mu}_k^i$, can be described as in Eq. (28).

$$\begin{aligned} \hat{\mu}_k^j &= \omega_0 + \sum_{j=0}^{J-1} \omega_i \varphi_k^j(\bar{\mu}_k^j) \\ &= \xi\left(\sum_{j=0}^{J-1} \varphi_k^j(\bar{\mu}_k^j)\right) = \xi\left(\sum_{j=0}^{J-1} \exp\left(-\frac{\left\|\bar{\mu}_k^j - d^j\right\|^2}{2(\tau^j)^2}\right)\right) \\ & (0 \le j \le J-1) \end{aligned} \tag{28}$$

$\bar{\mu}_k$ is an *n*-dimensional input vector and d^j stands for the center of the *j-th* basis function with the same dimension of the input vector. In the equations considered, τ^j denotes the width of the basis function, N is the number of hidden layer's nodes, $\left\|\bar{\mu}_k^j - d^j\right\|$ describes the Euclidean norm of $\bar{\mu}_k^j - d^j$ representing the distance between $\bar{\mu}_k^j$ and d^j, and $\varphi^j(x)$ means the response of the *j-th* basis function of the input vector with a maximum value at d^j.

The next process to obtain the prior mean and the covariance is to update the results from Eq. (27) or Eq. (28). The process described in the above 5 steps repeats until the end of the navigation.

$$\mu_{k+1} = \hat{\mu}_{k+1} + K_{k+1}^i (z_{k+1} - \hat{z}_{k+1}^i) \tag{29}$$

$$\Sigma_{k+1} = (I - K_{k+1}^i H_{k+1}^i)\bar{\Sigma}_{k+1} \tag{30}$$

4. Simulations

To show the effectiveness of the proposed algorithm, the Matlab code, developed by Bailey,[23] was modified. The simulation was performed with constraints on velocity, steering angle, system noise, observation noise, etc., for a robot with a wheel diameter of 1[m] and maximum speeds of 3[m/sec]. The maximum steering angle and speed are 25[°] and 15[°/sec] respectively. The control input noise is assumed to be a zero mean Gaussian with σ_v (=0.2[m/s]) and σ_φ (=3[°]). For observation, the number of arbitrary features around waypoints was used. In the observation step, a range-bearing sensor model and an observation model were used to measure the feature position and robot pose, which includes a noise with level of 0.1[m] in range and 1[°] in bearing. The sensor range is restricted to 15[m] for reducing operations, which is sufficient to detect all features in front of the mobile robot.

In this research, three navigation cases of the robot are surveyed: a linear navigation, a rectangular navigation, and a circular navigation. Specifications of the navigation maps are described in Table 1.

Table 1 Fundamental specification for navigation

Item	Horizontal	Rectangular	Circular
Feature	20	40	29
Waypoint	5	5	5
Area[m]	40 × 40	30 × 30	30 × 35

4.1 Navigation on Linear map

In the linear navigation case, as shown in Fig. 3, the robot navigates from the lower left corner to the upper right corner. The solid line depicts the robot's path, and the dash-dot line describes thea robot trajectory based on the data by real odometry. In addition, the star marks represent landmarks, and the cross marks are the covariance.

In Fig. 4, the dot lines, the dash-dot lines and the solid lines are the x, y, and heading angle errors in the case of EKF SLAM, UKF SLAM and Hybrid filter SLAM with MLP algorithm, respectively. In addition, the dashed lines show the results of the Hybrid filter SLAM with the RBF algorithm. They show the similar pattern both cases of EKF and UKF, and MLP and RBF.

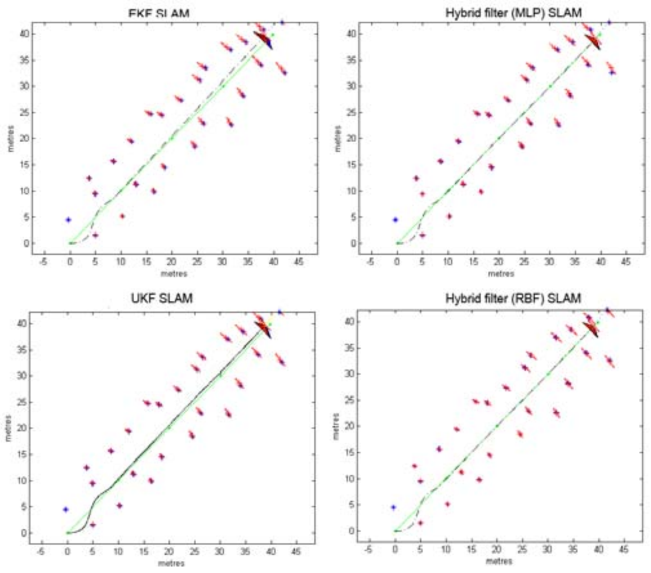


Fig. 3 Navigation result on linear map

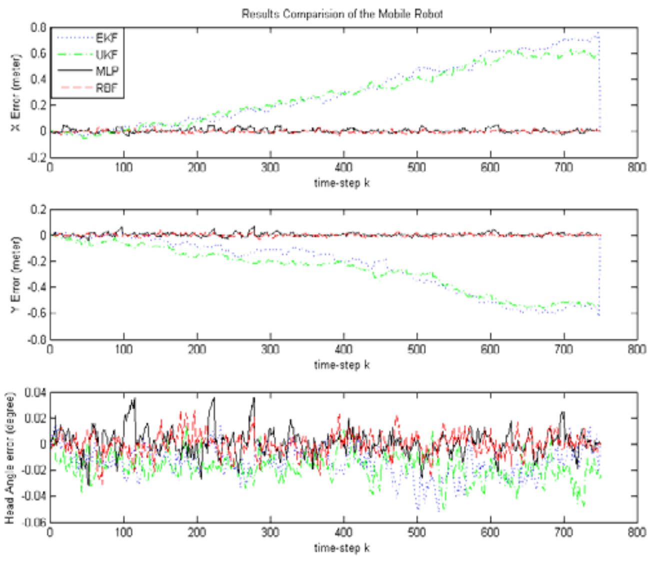


Fig. 4 Navigation errors on linear map

When a mobile robot navigates from the left-bottom to the right-top, the x-errors and y-errors have a similar pattern. Based on the simulation results, navigation using the Hybrid filter SLAM is more stable than other cases about the coordinate components, as shown in Fig. 5.

4.2 Navigation on Rectangular map

In the case of rectangular navigation, the EKF based navigation and Hybrid filter based navigation are shown in Fig. 5, where both results show distortions during navigation at the three edges. The mobile robot decides a direction for the navigation based on the information from the locations of landmarks detected, but it does not instantly turn because it has 1[°] in bearing when the robot tries to turn through the edges. In addition, the covariances during the navigation are different, as shown in Fig. 5. Estimating results (cross marks) of the features are found to be more reliable in the case of EKF SLAM since the original poses of the features have very similar positions to the estimated covariances.

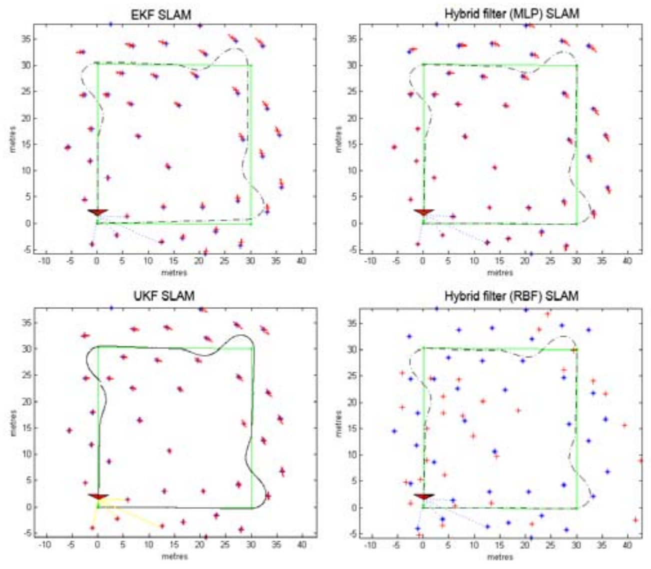


Fig. 5 Navigation result on rectangular map

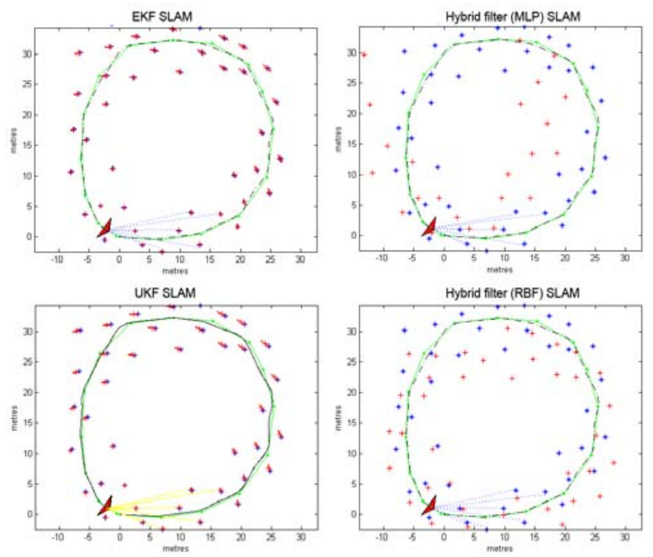


Fig. 7 Navigation result on circular map

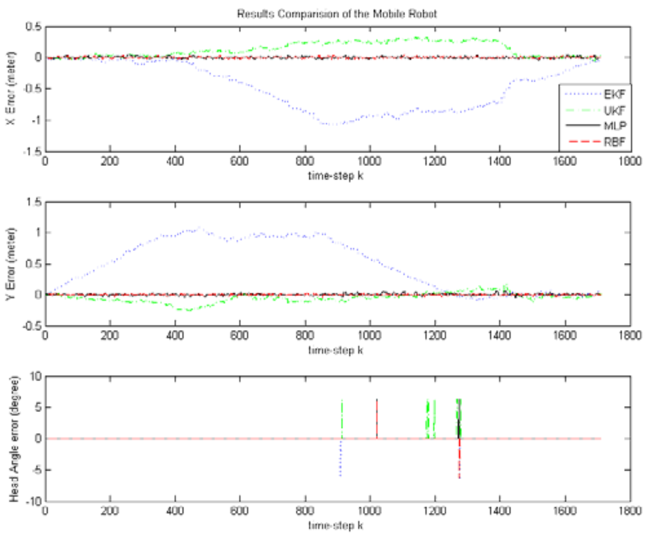


Fig. 6 Navigation errors on rectangular map

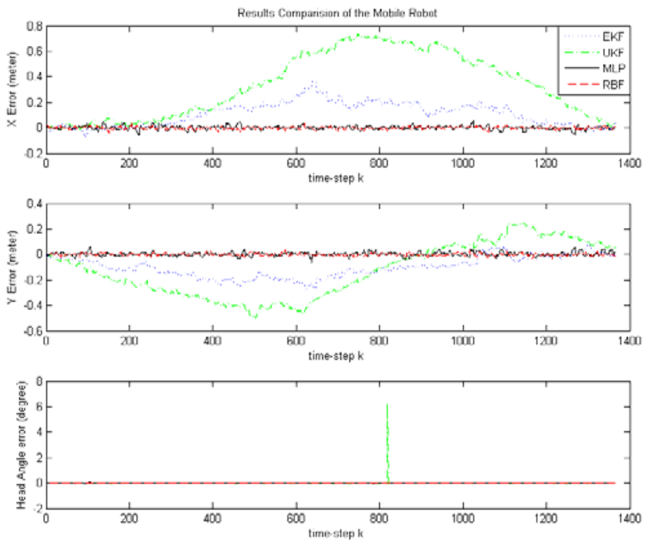


Fig. 8 Navigation errors on circular map

As shown in Fig. 6, until around 400th time steps, all results with different approaches shows similar results in the x-error. However, after the 400 time step, the RBF based SLAM shows the smallest error. The algorithm based on EKF SLAM results in the biggest error, but UKF SLAM results show better performance than EKF SLAM. In the y-axis direction, the RBF based algorithm shows the smallest error among the three approaches.

4.3 Navigation on Circular map

In the case of circular navigation, similar results to those of previous cases have been obtained. The best results are found to occur when the proposed Hybrid filter SLAM with the RBF algorithm is applied. The simulation results as shown in Fig. 8, the UKF based SLAM does not always show better performance than the EKF based SLAM. Through the total error, RBF based SLAM shows the most stable as shown in Fig. 8. Overall, the simulation results based on RBF and MLP show very similar pattern. Since MLP based approach required the longest process time, Hybrid filter SLAM with RBF was used in the experiment.

5. Experiments

To verify the proposed algorithm that was testified through simulation, some experiments are performed under two different conditions such as a standstill and straight navigation of the robot. In the navigation experiment, the robot equipped with three ultrasonic sensors and one CCD sensor navigates within the test ground with dimension of 6,500 x 2,100 mm under fluorescent lamp to compare the results using EKF SLAM with Hybrid filter SLAM. Through the navigation of the robot, the robot uses

Table 2 Specifications of the robot

Item	Specifications
Size(W×L×H)	120×140×90
Main processor	ATmega128
Vision sensor	CMOS(20pfs as VGA)
Ultrasonic sensor	SRF04
Communication	Firmteck 755A (Bluetooth)

Bluetooth communication for the information of SLAM input data and feedback results.

5.1 Fundamental Algorithm

During the navigation of the robot, the robot transmits images of the environment to server PC and disconnects with PC for securing resources such as controller which consists of Atmega series, drive motors and ultrasonic sensors. The measured information on encoder value of motors, distances between the front, left and right side of robot and the wall are sent to server PC for localization, mapping and display by using Bluetooth network. Based on the information, the server PC transmits new poses as result of SLAM algorithm to the mobile robot. The total process for experiment is shown in Fig. 9.

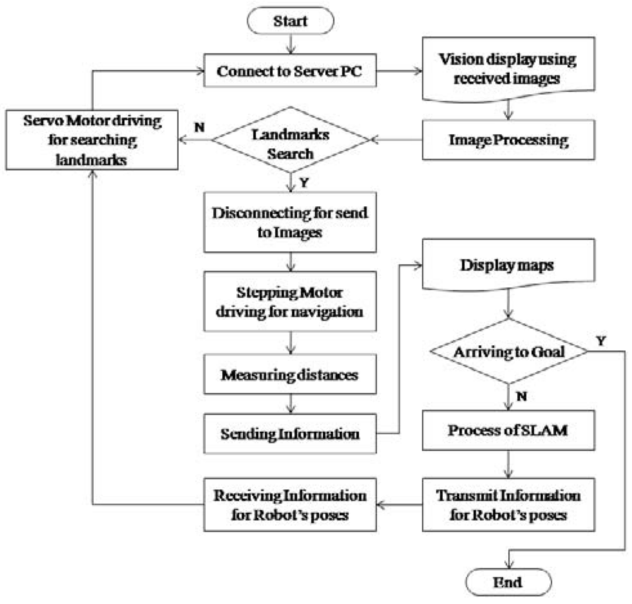
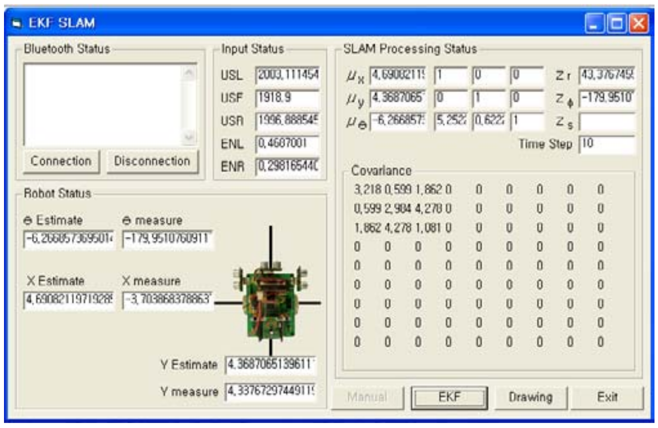

Start
Connect to Server PC
Vision display using received images
Servo Motor driving for searching landmarks
N
Landmarks Search
Image Processing
Y
Disconnecting for send to Images
Display maps
Stepping Motor driving for navigation
Arriving to Goal
Y
Measuring distances
N
Sending Information
Process of SLAM
Receiving Information for Robot's poses
Transmit Information for Robot's poses
End

Fig. 9 An experimental flowchart for SLAM

Experiments for SLAM of the robot with linear map using the EKF SLAM and the Hybrid filter SLAM are performed. For this experiment, we developed the SLAM program using Visual Basic 6.0 which operated for the SLAM algorithm because the micro-controller installed in the robot cannot operate sufficiently the SLAM algorithm.



Input Status
USL 2003,111454
USF 1918.9
USR 1996,888545
ENL 0,4607001
ENR 0,29816544C

SLAM Processing Status

μ_x 4,69082115 1 0 0 z_r 43,376745
μ_y 4.3687065 0 1 0 z_ϕ -179.9510
μ_θ -6,266857 5,252 0,622 1 z_s
Time Step 10

Covariance

```
3,218 0,599 1,862 0  0  0  0  0  0
0,599 2,904 4,278 0  0  0  0  0  0
1,862 4,278 1,081 0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
0     0     0     0  0  0  0  0  0
```

Robot Status
θ Estimate -6,2668573695014
θ measure -179,9510760911
X Estimate 4,69082119719288
X measure -3,703868378863
Y Estimate 4.3687065139611
Y measure 4.3376729744911

Fig. 10 SLAM processing simulator for an experiment

Axis x, y and a heading angle θ are the perpendicularity of the heading direction, the heading direction of the robot and an angular between axis x and the robot's heading, respectively. We basically assumed the robot knew the features' pose and accurate encoder values. Fig.10 shows the processing simulator developed with Visual Basic 6.0 for experiments of an actual environment. USL, USF, USR describe the measurement values in millimeter from three ultrasonic sensors posed on the left, front, and right side, respectively. The parameters in 'SLAM Processing status' show the means, the measurements, and the covariance with 3 x 3 matrix.

5.2 The Experimental under a Standstill Environment

Fig. 11 shows the experimental system to test the algorithm under a standstill environment where the robot goes back and forth.

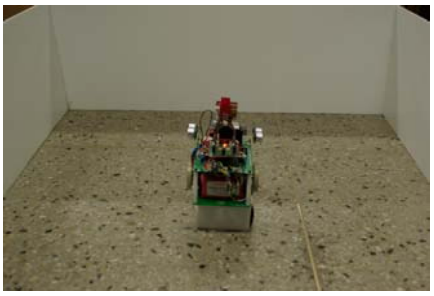


Fig. 11 The standstill environment to test the proposed algorithm

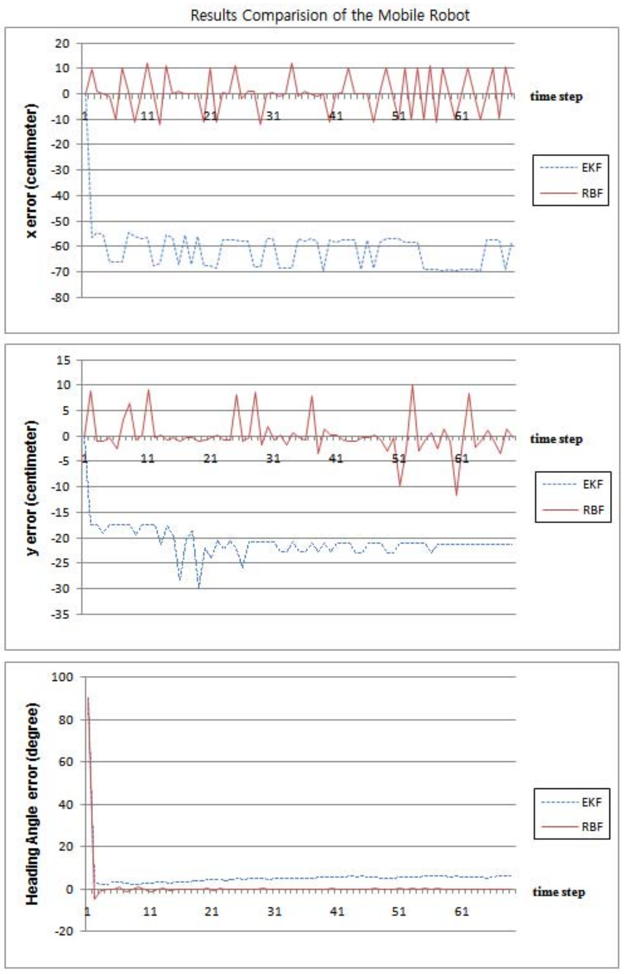


Fig. 12 Experimental results under a standstill environment

The experimental results as shown in Fig. 12 confirm the simulation results where the information on the environment is obtained using sets of ultrasonic sensors and two encoders of the robot wheel.

The experimental results as in Fig. 12 justify the proposed algorithm since they have very similar trend to the simulation results using Matlab. Beside the initial error of the robot angle, the results based on RBF show more stable performance than EKF case.

5.3 The Experimental for Navigation of the Robot

In case of linear navigation, the experimental results show similar pattern as in standstill case. When EKF SLAM is applied, the robot moves straight with zigzag pattern as shown in Fig. 13. The navigation patterns for RBF depict more stable results comparing with the case of EKF SLAM.

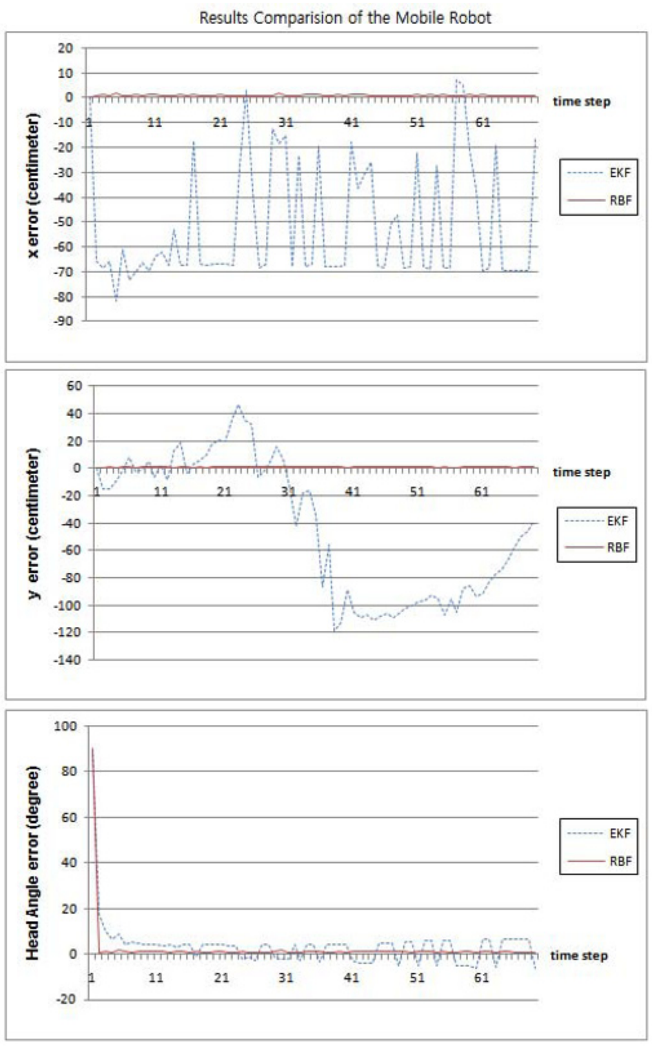


Fig. 13 Experimental results on the linear map

The navigation pattern of the robot shows the tendency of turning left on applying both of the algorithms because of the slip between the wheels and the floor. Fig. 14(a), (b) and (c) show the environment and the detecting result of features and navigation of the robot, respectively.

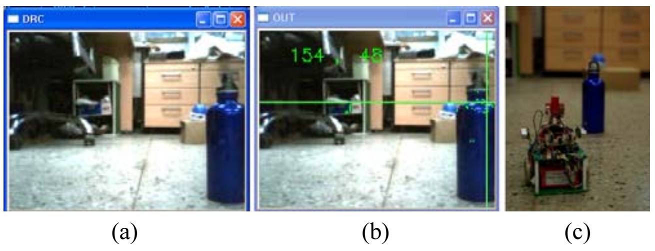


Fig. 14 The experimental process (a) The scene windowed through vision system, (b) Post- process with detecting landmarks, (c) Searching landmarks during the navigation

6. Conclusions

The SLAM is one of the most fundamental problems in the quest for autonomous mobile robots since the robot keeps track of its location by maintaining a map of the physical environment and an estimate of its position on that map. This paper proposes Hybrid filter SLAM methods, such as the MLP SLAM and the RBF SLAM with EKF on a mobile robot, to make up for the EKF SLAM error inherently caused by its linearization process and noise assumption. The proposed algorithm consists of two steps: the Neural Networks and the EKF algorithm. The simulation results for three different navigation cases show that the efficiency of the proposed algorithm based on RBF as compared with the EKF SLAM, in terms of parameters such as x, y, and θ. To verify the effectiveness of the proposed algorithm, simulation in Matlab with EKF, UKF, RBF and MLP with EKF are performed. Based on the simulation results, EKF and UKF based SLAM show very similar results, but EKF SLAM results in the biggest errors. In addition, the MLP based approach required around 3 times of the process time comparing with other algorithms.

Through the simulation and experimental results for the EKF SLAM and Hybrid filter SLAM with RBF, the effectiveness of the proposed algorithm is verified. The experimental results show the same tendency of the simulation ones. In addition, the results confirm the Hybrid filter SLAM is more stable for robot navigation in both the simulation and experiment.

Research under harsh and real-time condition is under way to verify the robustness of the proposed algorithm by changing structures of neural networks.

ACKNOWLEDGEMENT

This research was supported by the Yeungnam University research grants in 2008.

REFERENCES

1. Kim, J. M., Kim, Y. T. and Kim, S. S., “An accurate localization for mobile robot using extended Kalman filter and sensor fusion,” IEEE International Joint Conference on Neural Networks, pp. 2928-2933, 2008.

2. Lee, S. J., Lim, J. H. and Cho, D. W., "Effective Recognition of Environment Using Sonar Ring Data for Localization of a Mobile Robot," Proceedings of the Korean Society of Precision Engineering Spring Conference, pp. 1-2, 2008.

3. Kim, G. S., "Perception of small-obstacle using ultrasonic sensors for a mobile robot," Proceedings of the Korean Society of Precision Engineering Autumn Conference, pp. 21-24, 2004.

4. Panzieri, S., Pascucci, F. and Setola, R., "Multirobot localisation using interlaced extended Kalman filter," IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2816-2821, 2006.

5. Caron, F., Davy, M., Duflos, E. and Vanheeghe, P., "Particle Filtering for Multisensor Data Fusion With Switching Observation Models: Application to Land Vehicle Positioning," IEEE Transactions on Signal Processing, Vol. 55, Issue 6, pp. 2703-2719, 2007.

6. Lee, S. J., Lim, J. H. and Cho, D. W., "EKF Localization and mapping by using consistent sonar feature with given minimum features," SICE-ICASE International Joint Conference, pp. 2606-2611, 2006.

7. Houshangi, N. and Azizi, F., "Accurate mobile robot position determination using unscented Kalman filter," 2005 Canadian Conference on Electrical and Computer Engineering, pp. 846-851, 2005.

8. Zhu, J., Zheng, N., Yuan, Z., Zhang, Q. and Zhang, X., "Unscented SLAM with conditional iterations," 2009 IEEE Intelligent Vehicles Symposium, pp. 134-139, 2009.

9. Harb, M., Abielmona, R., Naji, K. and Petriul, E., "Neural networks for environmental recognition and navigation of a mobile robot," IEEE International Instrumentation and Measurement Technology Conference, pp. 1123-1128, 2008.

10. Zu, L., Wang, H. K. and Yue, F., "Artificial neural networks for mobile robot acquiring heading angle," Proceedings of the Third Intemational Conference on Machine Laming and Cybemetics, pp. 26-29, 2004.

11. Bugeja, M. K. and Fabri, S. G., "Multilayer Perceptron Adaptive Dynamic Control for Trajectory Tracking of Mobile Robots," IEEE Industrial Electronics Annual Conference, pp. 3798-3803, 2006.

12. Vafaeesefat, A, "Optimum Creep Feed Grinding Process Conditions for Rene 80 Supper Alloy Using Neural network," Int. J. Precis. Eng. Manuf., Vol. 10, No. 3, pp. 5-11, 2009.

13. Cho, S. H., "Trajectory Tracking Control of a Pneumatic X-Y Tabel using Neural Network Based PID Control," Int. J. Precis. Eng. Manuf., Vol. 10, No. 5, pp. 37-44, 2009.

14. Choi, M. Y., Sakthivel, R. and Chung, W. K., "Neural network-aided extended Kalman filter for SLAM problem," IEEE International Conference on Robotics and Automation, pp. 1686-1690, 2007.

15. Stubberud, S. C., Lobbia, R. N. and Owen, M., "An Adaptive Extended Kalman Filter Using Artificial Neural Networks," Proceedings of the 34th Conference on Decision &Control, pp. 1852-1856, 1995.

16. Hu, Y. H. and Hwang, J. N., "Handbook of Neural Network Signal Processing," CRC Press, pp. 3.1-3.23, 2001.

17. Jang, P. S., "Neural network based position tracking control of mobile robot," M.S thesis, Department of Mechatronics, Chungnam National University, pp. 13-37, 2003.

18. Oh, C. M., "Control of mobile robots using RBF network," M.S thesis, Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, pp. 4-19, 2003.

19. Mehra, P. and Wah, B. W., "Artificial neural networks: Concepts and theory," IEEE Computer Society Press, pp. 13-31, 1992.

20. Iiguni, Y., Sakai, H. and Tokumaru, H., "A Real-Time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter," IEEE Transactions on Signal Processing, Vol. 40, No. 4, pp. 959-966, 1992.

21. Thrun, S., Burgard, W. and Fox, D., "Probabilistic Robotics," The MIT Press, pp. 309-334, 2005.

22. Bailey, T., Nieto, J., Guivant, J., Stevens, M. and Nebot, E., "Consistency of the EKF-SLAM Algorithm," IEEE International Conference on Intelligent Robotics and Systems, pp. 3562-3568, 2006.

23. Bailey, T., http://www-personal.acfr.usyd.edu.au/tbailey.