TELKOMNIKA, Vol.13, No.1, March 2015, pp. 93~102
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v13i1.1275

Hierarchical-map Updating Approach for Simultaneous Localization and Mapping of Mobile Robots

Yingmin Yi*, Zhimin Wang
Faculty of Automation and Information Engineering, Xi'an University of Technology,
Xi'an710048, Shaanxi, China
*Corresponding author, e-mail: yiym@xaut.edu.cn

Abstract

For the tremendously increasing of system state in wild field, the computational complexities of mobile robot system should be taken into account. This paper proposes a hierarchical-map updating approach for simultaneous localization and mapping of robots. The basic idea of hierarchical-map is defining two kinds of maps during the recursive updating process, namely local map (upper map) and global map (lower map). The system states will be updated by the preset maps. The hierarchical-map updating process is just for the upper map and the lower map is updated after a certain running term. In the calculation, the state data of the upper map is far less than that of the lower map. It is validated by the experiments that, the approach is more optimal than others in computational complexities while ensuring the consistency estimate.

Keywords*: hierarchical-map updating, computational complexity, simultaneous localization and map building, mobile robots*

1. Introduction

The moving and map building in wild environment is of great importance in the research of robot filed. Due to the large area, the system states increase exponentially. The researches for robot SLAM follow the variations of the application environments, from 2-dimentional environment [1] to 3-dimentional environment [2], from structured environment [3] to none-structured environment [4], from indoor [5] to outdoor [6] and other natural none-structured environment [7]. For the SLAM of vehicle robots, it is necessary to update the pose and map after every new observation [8].

With the increasing of the features on the map, the computational complexity increases as well. For the computational problem, researchers put forward several map-division approaches [9]. The essence of the map-division approaches is to reduce the times of updating the global map while ensuring the optimal estimate. The map-division approaches have two categories: one is operating on the local parts of the map, remaining the global reference coordinates. The Compressed Extended Kalman Filtering (CEKF) proposed in paper [10], addresses the local area data combing approach, lowering the computational complexity of the algorithm; the other is the sub-map technology under the local coordinate frames. The every-increasing problem of matrixes is discussed in [11].

The EKF approach for SLAM is based on the minimum mean square error, completing the optimal recursion for robot poses in time domain. But for each updating, the whole matrix is processed, increasing the computational complexity, which greatly restricts the approach applying in the large-scale environment. For the ever-increasing problem of the EKF-SLAM with the increment of the states, this paper proposes a hierarchical-map updating approach for SLAM (HMU-SLAM). During the recursion process of the robot, the state space is divided to two layers, namely the lower-map (global) state space and upper-map (local) state space. In the SLAM algorithm, at every recursion step, the updating step is just for updating the upper state space. When the robot navigates out the upper-map area, then the lower-map is updated. This algorithm can lower the computation dimensions effectively so as to lower the computational complexity, making it possible of its implementation in large-scale environment.

The next section presents the process and observation models used in our experiments, which sets the context for these results in terms of a 2-D vehicle with a range bearing sensor. The landmark model is discussed in the section. Section III describes hierarchical-map updating approach for simultaneous localization and mapping of robots.

Received October 13, 2014; Revised December 21, 2014; Accepted January 8, 2015

Section IV presents the results of the simulation experiments in terms of computational complexities and estimate consistency. The final section provide discussion and conclusions.

2. System model

2.1. SLAM system model

The described SLAM system is composed by robot's position and direction and the observed coordinates on static environment landmark. The united state vector under k state is shown as

$$\mathrm{x}_k = \begin{bmatrix} x_{vk} \\ y_{vk} \\ \theta_{rk} \\ x_1 \\ y_1 \\ \vdots \\ x_N \\ y_N \end{bmatrix} = \begin{bmatrix} \mathrm{x}_{vk} \\ \mathrm{n} \end{bmatrix} \tag{1}$$

In the formula, $x_{vk}, y_{vk}, \theta_{rk}$ represent respectively the robot's coordinates in two-dimensional space and direction angle. The map is static, parameter $\mathrm{n} = [x_1, y_1, \cdots, x_N, y_N]^{\mathrm{T}}$ has no time index. The robot's movement model is rolling motion constraints (i.e., assuming zero wheel slip) [12].

$$\mathrm{x}_k = f_v(\mathrm{x}_{vk-1}, \mathrm{u}_k) = \begin{bmatrix} x_{vk-1} + v_k \Delta T \cos(\theta_{rk-1} + G_k) \\ y_{vk-1} + v_k \Delta T \sin(\theta_{rk-1} + G_k) \\ \theta_{rk-1} + \frac{v_k \Delta T}{B} \sin(G_k) \end{bmatrix} \tag{2}$$

The time interval from $k-1$ to k moment is ΔT, speed v_k and driving angle G_k are constant, which constitutes controlled quantity $\mathrm{u}_k = [v_k, G_k]^{\mathrm{T}}$, robot's wheel base is B.

2.2. SLAM observation model

The observed model is where $\mathbf{z}_{ik}$ is the observation vector at time k and h_i is the model of the observation of the i th landmark. The vector $\mathbf{z}_{ik}$ is a observation of the landmark location $\mathbf{p}_i$ relative to the robot's location $\mathbf{x}_{vk}$. This type of observation will be referred to as a vehicle-landmark observation or a VLM observation.

$$\mathrm{z}_{ik} = h_i(\mathrm{x}_k) = \begin{bmatrix} \sqrt{(x_i - x_{vk})^2 + (y_i - y_{vk})^2} \\ \arctan \frac{y_i - y_{vk}}{x_i - x_{vk}} - \theta_{rk} \end{bmatrix} \tag{3}$$

The model is not assumed to be perfect and unmodelled sensor characteristics and noise corruption are lumped into a observation error vector ω_{ik}. The observation error vector is again taken to be a temporally uncorrelated and zero mean random sequence.

2.3. Landmark model

Landmarks are fixed and conspicuous features within the environment. Landmarks can have many physical forms; corners, planes, rough surfaces, poles, natural or artificial terrain features can all be considered landmarks if they are repeatedly and reliably observed by a

sensor. Exactly what constitutes a landmark is driven by the physics of the observing sensor - landmarks are conspicuous through the eyes of the observing sensor. This sensor-centric definition of a landmark means that it is not always possible to readily associate a landmark with visually perceived features.

Mathematically, landmarks are represented as a vector of parameters that define the location and other properties of the landmark. This thesis generally employs the simplest of all landmark models: a landmark is a stationary point like entity in two dimensions. A point landmark is defined by two parameters specifying its position in cartesian space with respect to some global coordinate frame. A point landmark is visible from all viewing angles. In general, more complex landmarks can be incorporated into the maps and models employed throughout this thesis.

The i th point landmark in the environment will be denoted as $\mathbf{p}_i$ and will be defined as follows

$$\mathbf{p}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}^{\mathrm{T}} \tag{4}$$

The relationship between the point landmark state at times k + 1 and k is trivial.

The landmark is stationary and so

$$\mathbf{p}_i(k+1) = \mathbf{p}_i(k) = \mathbf{p}_i \tag{5}$$

Importantly, and in contrast to the vehicle model, there is no additive uncertainty term in the landmark model.

The vehicle motion model, the observation model, and the measured values of the control parameters $\mathrm{u}_k = [v_k, G_k]^{\mathrm{T}}$, are not exact, but are subject to noise, which lead to uncertainty in the state estimate. For this reason, we require a probabilistic filter to recursively estimate a distribution over the state given noisy information.

3. Hierarchical-map updating algorithm for SLAM

3.1. Basic ideas of hierarchical-map

When the robot conducts SLAM in the wild environment, the map data and the system state space will increase exponentially. When the robot is working in the wild, what concerns are the interested areas around and the general objects. Hence, for the map building of the robot, it is available to divide the interested areas and the general objects to hierarchical maps. According to the requirements, the map can be divided into N layers. The top layer is denoted by 1. And the bottom layer is denoted by N. The schematic graph of the hierarchical maps is shown in Figure 1.

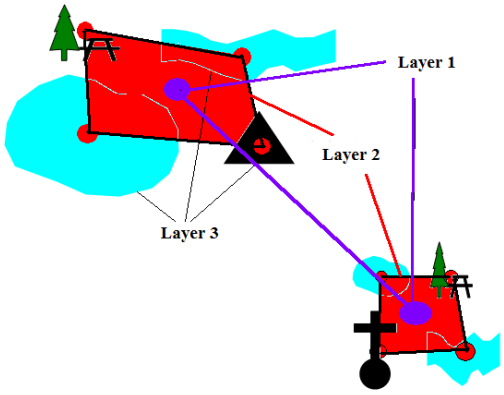

Figure 1. Schematic graph of the hierarchical maps

3.2. Updating for the hierarchical maps

For the problem of computational complexity increase resulted by the increment of the state space, the system area is divided into two parts during the EKF recursion process, namely the upper-map state space and the lower-map state space. During each step of the SLAM recursion step, the updating step just updates the upper-map state space. When the robot navigates out of the upper-map area, then the lower-map is updated. As it is shown in Figure 2, the rectangular Area A is the initial upper-map state space; Area B is the initial lower-map state space. When the sensor of the mobile robot observes the features of the lower-map state space while navigating in the upper-map state space, the lower map will be updated. Besides, the upper-map state space and the lower-map state space are re-divided, as shown in Figure 2.

The HMU-SLAM algorithm is described as follows:

Initialisation and prediction: the upper-map area is independent from the lower-map area. The state vectors are divided to two parts.

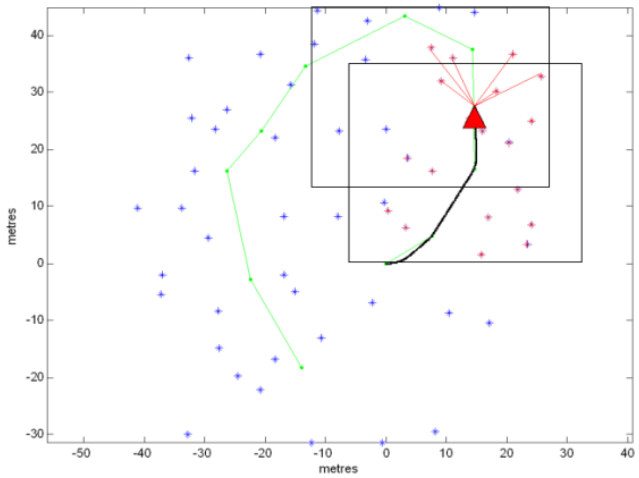

Figure 2. Updating for the state spaces

$$\boldsymbol{X} = \begin{pmatrix} X_{\text{Top}} \\ X_{\text{Low}} \end{pmatrix}, X_{\text{Top}} \in R^{n_{\text{Top}}}, X_{\text{Low}} \in R^{n_{\text{Low}}} \tag{6}$$

where, n presents the number of the virtual features. The covariance matrixes are divided as follows:

$$\boldsymbol{P} = \begin{pmatrix} P_{\text{TT}} & P_{\text{TL}} \\ P_{\text{LT}} & P_{\text{LL}} \end{pmatrix} = E\left\{(\overline{X} - X)(\overline{X} - X)\right\} \in R^{n} \tag{7}$$

The auxiliary matrix: $\phi, \psi \in R^{n_{\text{Top}}, n_{\text{Low}}}, \theta \in R^{n_{\text{Top}}}, Q^{*}_{\text{LL}} \in R^{n_{\text{Top}}, n_{\text{Low}}}$

The initial time k_1:

$$\phi(k_1) = I, \psi(k_1) = 0, \theta(k_1) = 0, Q^{*}_{\text{LL}}(k_1) = 0 \tag{8}$$

Different from the EKF-SLAM algorithm, the HMU-SLAM algorithm conducts the prediction in the upper-map area. At this time, only $X_{\text{Top}}, P_{\text{TT}}$ are involved in the prediction. The auxiliary matrix has a recursive equation as the following.

$$\phi(k) = J_{\text{TT}}\phi(k-1) \tag{9}$$

$$\psi(k) = \psi(k-1) \tag{10}$$

$$\theta(k) = \theta(k-1) \tag{11}$$

$$J_{\mathrm{TT}} = (\partial f_{\mathrm{Top}} / \partial X_{\mathrm{Top}}) \mid X_{\mathrm{Top}}(k) \tag{12}$$

$$Q^{*}_{\mathrm{LL}}(k) = Q^{*}_{\mathrm{LL}}(k-1) + Q_{\mathrm{LL}}(k) \tag{13}$$

Updating: the updating is divided to upper-map updating and lower-map updating. The upper-map updating is just for the local areas and the lower-map updating is conducted via the recursion of the auxiliary matrix.

$$\phi(k) = (I - \mu(k)\phi(k-1)) \tag{14}$$

$$\beta(k) = H^{\mathrm{T}}_{\mathrm{Top}}(k) S(k)^{-1} H_{\mathrm{Top}}(k) \tag{15}$$

$$\psi(k) = \psi(k-1) + \phi^{\mathrm{T}}(k-1)\beta(k)\phi(k-1) \tag{16}$$

$$Z(k) = y(k) - h(X_{\mathrm{Top}}(k), k) \tag{17}$$

$$S(k) = H_{\mathrm{Top}}(k) P_{\mathrm{TT}}(k) H^{\mathrm{T}}_{\mathrm{Top}}(k) + R \tag{18}$$

$$H_{\mathrm{Top}}(k) = (\partial h / X_{\mathrm{Top}}) \mid X_{\mathrm{Top}}(k) \tag{19}$$

$$\mu_k = P_{\mathrm{TT}}(k)\beta(k) \tag{20}$$

$$\theta(k) = \theta(k-1) + \phi^{\mathrm{T}}(k-2) H^{\mathrm{T}}_{\mathrm{Top}}(k-1) S(k-1)^{-1} Z(k-1) \tag{21}$$

Updating of $X_{\mathrm{Low}}, P_{\mathrm{TL}}, P_{\mathrm{LL}}$: Once the area of the lower state space is observed (at time K2 here), the lower-map updating is conducted via the recursion formula.

$$P_{\mathrm{TL}}(k_2) = \phi(k_2) P_{\mathrm{TL}}(K_1) \tag{22}$$

$$P_{\mathrm{LL}}(k_2) = P_{\mathrm{LL}}(k_1) - P_{\mathrm{TL}}(k_1)\psi(k_2) P_{\mathrm{TL}}(k_1) + Q^{*}_{\mathrm{LL}}(k_2) \tag{23}$$

$$X_{\mathrm{Low}}(k_2) = X_{\mathrm{Low}}(k_1)\theta(k_2) \tag{24}$$

3.3. Steps for HMU-SLAM

HMU-SLAM process is performed as follows.

Step1 Initialisation: Initially define and assign values for the prediction location, system covariance matrixes, hierarchical areas, the auxiliary matrix, the system process noise covariance matrix, the system observation noise covariance matrix, control variables, the maximum observation distance and the time interval.

Step2 Location prediction for the robot: predict the current location and covariance matrix of the robot according to the position estimate, covariance matrix, auxiliary matrix and moving model of the former moment in the upper-map area.

Step3 Practical observation: Obtain the practical observations by the observations of the environment features using the sensors of the robot.

Step4 Prediction observation: Calculate the prediction observations of the environment features using the measurement model according to the coordinates of the location prediction and observation environment features on the lower map.

Step5 Data association (relating): conduct the data association for the observation features of the upper map.

$$Inn_k = z(k) - h(\hat{x}_{k|k-1}, \hat{m}_{k-1}) \quad (25)$$

$$D_k = Inn_k^{\mathrm{T}} S_k^{-1} Inn_k \quad (26)$$

Step6 State updating: the state updating is divided to upper-map updating and lower-map updating. The upper-map updating is just for the associated features. The data is generally far less than that of the lower map.

The lower-map updating is generally conducted after a certain distance of movement of the robot, to modify the data of the global map.

Step7 State incrementing: Add the new observed features to the map. The HMU-SLAM algorithm is recursive by steps in sequence of prediction, observation, data association, upper-map updating or lower-map updating and state incrementing.

$$\mathbf{x}_k^{new} = h^{-1}(\mathbf{z}_{nk}, \mathbf{x}_{vk}) \quad (27)$$

$$\mathbf{x}_k = \begin{bmatrix} \mathbf{x}_k \\ \mathbf{x}_k^{new} \end{bmatrix}^{\mathrm{T}} \quad (28)$$

4. Results and Analysis

The initial state and covariance matrix of the experiments are $\hat{x}_{0/0} = zeros(3,1)$, $P_{0/0} = zeros(3)$, respectively. In the initialisation process, the noise covariance and the observation covariance are $Q = diag[0.3^2, (3\pi/180)^2]$, $R = diag[0.1^2, (\pi/180)^2]$,respectively.The experiments also performed the classic EKF-SLAM, FastSLAM, UKFSLAM , for comparing with and analyzing the proposed algorithm. Figure 3 shows classical EKF-SLAM algorithm. Figure 4 shows classical Fast-SLAM algorithm. Figure 5 shows the hierarchical-map SLAM of the proposed algorithm. Figure 6 shows the two-layer map SLAM. The map is built by using the proposed algorithm.

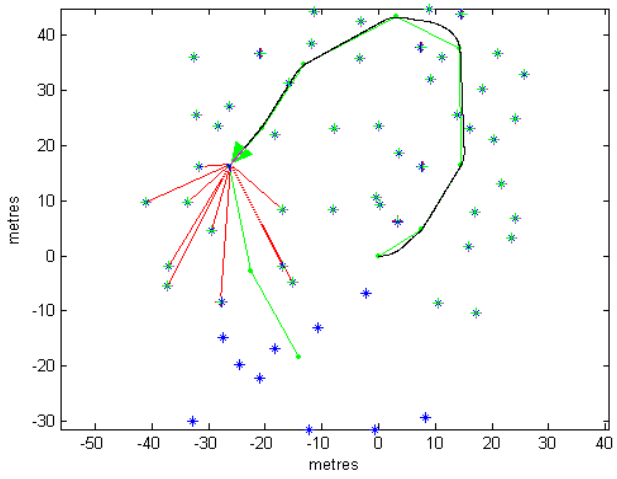

Figure 3. The map of the robot EKF-SLAM

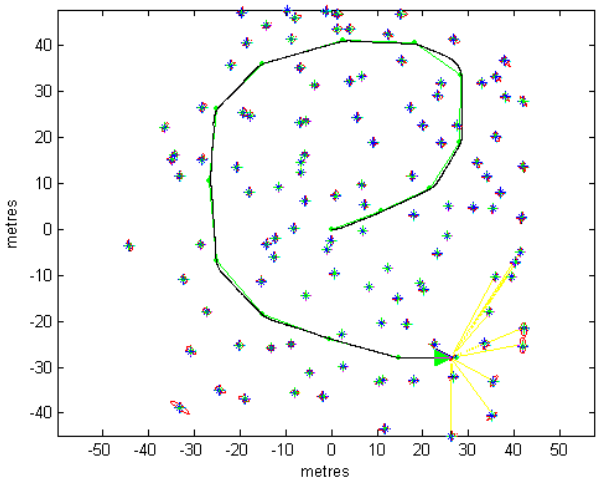

Figure 4. The map of the robot Fast-SLAM

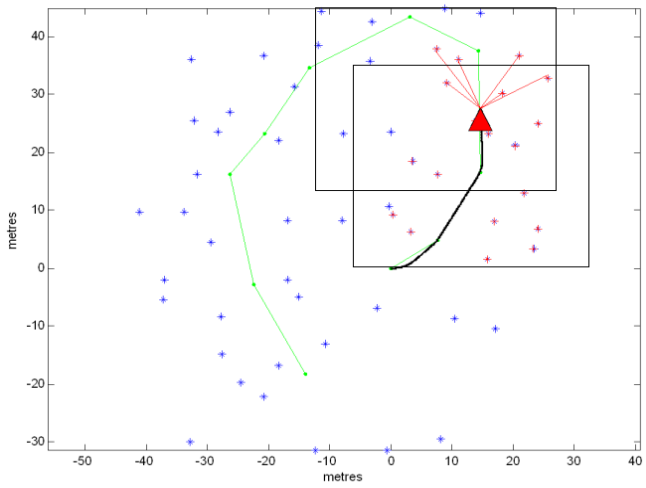

Figure 5. State space updating gram

The data processing quantity of a single-step calculation is an important index for deciding the efficiency of an algorithm [10]. For the large-scale map, to verify the computation complexity of the algorithm, the experiments also performed the classic EKF-SLAM, FastSLAM and UKFSLAM respectively in the same environment, for comparing with the proposed algorithm in this paper. Figure 7 shows the time consuming averages of the above algorithms under 50 single-step calculations. i presents step length. T presents time (unit: s).

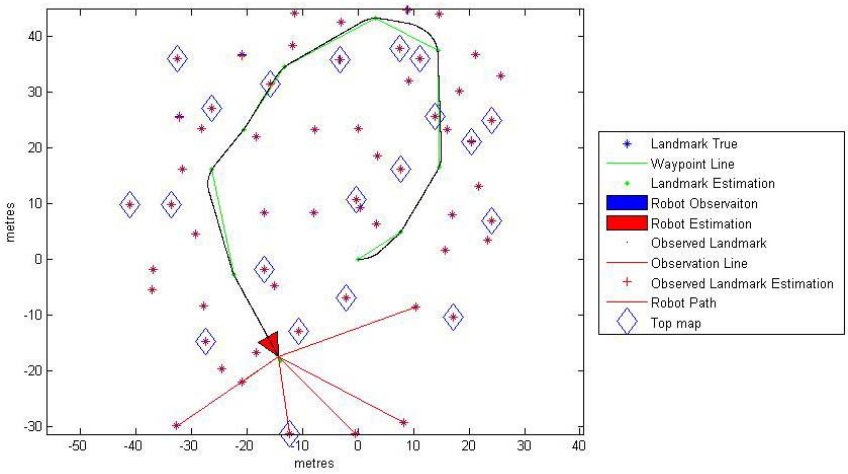

Figure 6. Hierarchical mapping of robot SLAM

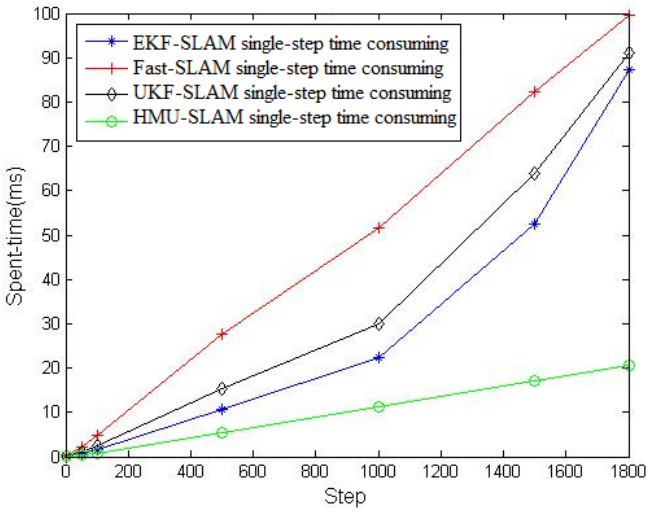

Figure 7. Comparision for single-step consusing of the algorithms

As it can be seen from Figure 7, under the same conditions, the single-step time consuming curve of the FastSLAM algorithm goes top; UKFSLAM is inferior; EKFSLAM is beneath UKFSLAM; HMUSLAM is at the bottom. This indicates the proposed algorithm has minimum single-step time consuming, optimal than the other algorithms in computational complexity.

For the proposed algorithm, the estimation issue of consistency should be considered. For linear Gaussian filter, the filter performance can be characterized through NEES (normalized estimation error squared) [11].

$$\varepsilon_k = (\mathrm{x}_k - \hat{\mathrm{x}}_{k|k})^{\mathrm{T}} \mathrm{P}_{k|k}^{-1} (\mathrm{x}_k - \hat{\mathrm{x}}_{k|k}) \tag{29}$$

Under the hypothesis that the filter is consistent and approximately linear-Gaussian, NEES obeys χ^2 distribution. Consistency of the algorithm is evaluated by performing N times Monte Carlo runs. The algorithm performance indicators are evaluated by the average NEES. When $N \to \infty$, $\bar{\varepsilon}_k$ approaches to the state vector dimension.

$$\bar{\varepsilon}_k = \frac{1}{N}\sum_{i=1}^{N}\varepsilon_{ik} \tag{30}$$

Given the hypothesis of a consistent linear-Gaussian filter, $N\bar{\varepsilon}_k$ has a χ^2 density with N dim (x_k). Thus, for the 3-dimensional robot pose, with N=50, the 95% probability concentration region for $\bar{\varepsilon}_k$ is bounded by the interval [2.36, 3.72]. If $\bar{\varepsilon}_k$ rises significantly higher than the upper bound, the filter is optimistic, if it tends below the lower bound, the filter is conservative.

To verify the consistency estimate, as to EKF-SLAM, FastSLAM, UKFSLAM and the proposed algorithm, we conducted 50 MonteCarlo runs on the robot for each. Figure 8 shows the NEES consistency estimate curves. As it can be seen from Figure 8, the NEES curves of the algorithms are almost within [2.36, 3.72]. Hence, they are all consistency estimates. From the above, the proposed algorithm is optimal in computational complexity while ensuring the consistency estimate.

5. Conclusion

For the problem of the ever-increasing computational complexity resulted by the ever-increasing state space in the large-scale environment, this paper proposes a hierarchical-map updating algorithm for simultaneous localization and mapping. According the algorithm, the robot will build multiple map layers at map building. During state updating, only a few states of the upper map are updated, thus lowering the computational complexity of the algorithm. The experiments validated that the proposed algorithm has the minimum computational complexity while ensuring the consistency estimate. The ideas of this paper provide a meaningful clue for the map building of the large-scale unknown environment for robot.

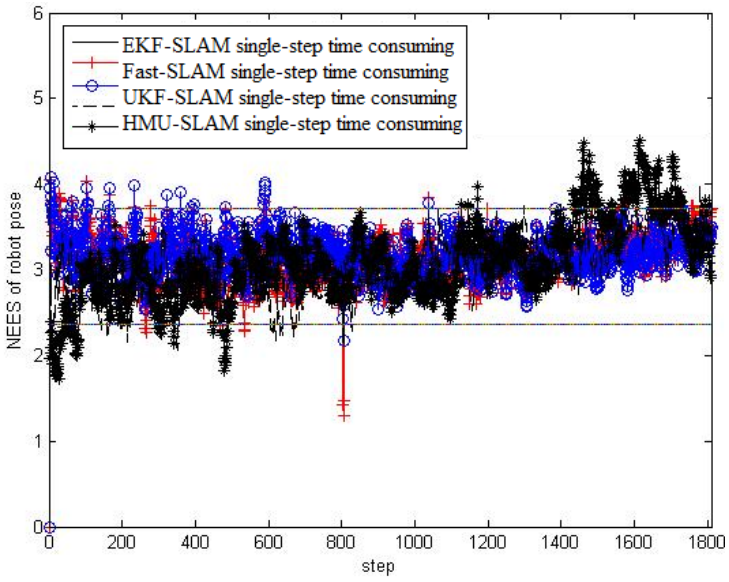

Figure 8. NEES consistency estimate curves of the algorithms

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 51275405) and the science research programs of education department of Shaanxi Province (2013JK1078).

References

[1] Yang P. Efficient particle filter algorithm for ultrasonic sensor-based 2D range-only simultaneous localisation and mapping application. *IET Wireless Sensor Systems.* 2012; 2(4): 394 -401.

[2] Bosse M, Zlot R, Flick PZ. Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping. *IEEE Transactions on Robotics.* 2012; 28(5): 1104-1119.

[3] Mitch B, Salah S. Architectures for Cooperative Airborne Simultaneous Localisation and Mapping. *Journal of Intelligent and Robotic Systems.* 2009; 55(4): 267-297.

[4] Weizhen Z, Miro JV, Dissanayake G. Information-Efficient 3-D Visual SLAM for Unstructured Domains. *IEEE Transactions on Robotics.* 2008; 24(5): 1078-1087.

[5] S Thrun, D Fox, W Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning.* 1998; (31): 29-53. also appeared in *Autonomous Robots* 5. 1998; 253-271 (joint issue).

[6] Jose G, Eduardo N, Stephan B. Autonomous Navigation and Map building Using Laser Range Sensors in Outdoor Applications. *Journal of Robotic Systems.* 2000; 17(10): 565-583.

[7] Jose G, Eduardo N, Juan N, Favio M. Navigation and Mapping in Large unstructured Environments. *The International Journal of Robotics Research.* 2004; 23(4): 449-472.

[8] Si-Yao F, Xin-Kai K, Rui Z, Guo-Sheng Y, Zeng-Guang H. *Compressive sensing approach based mapping and localization for mobile robot in an indoor wireless sensor network*. 2010 International Conference on Networking, Sensing and Control. Chicago,USA, IEEE. 2010: 122-127.

[9] Bai-Fan C, Zi-Xing C, Zhi-Rong Z. *A hybrid data association approach for mobile robot SLAM*. 2010 International Conference on Control Automation and Systems. Gyeonggi-do, Korea, IEEE. 2010: 1900-1903.

[10] Jose G, Eduardo N. *Improving Computational and Memory Requirements of Simultaneous Localization and Map Building Algorithms*. Proceedings of the 2002 IEEE international Conference on Robotics 8 Automation. Washington, DC, IEEE. 2002: 2731-2736.

[11] Y Bar-Shalom, XR Li, T Kirubarajan. *Estimation with applications to tracking and navigation*. John Wiley and Sons. 2001: 234-235.

[12] R Smith, M Self, P Cheeseman. A stochastic map for uncertain spatial relationships. *In International Symposium of Robotics Research.* 1987: 467-474.