Adaptive Computation Algorithm for Simultaneous Localization and Mapping (SLAM)

Da-Wei Kung, Chen-Chien Hsu, Wei-Yen Wang and Jacky Baltes

Abstract Computationally Efficient SLAM (CESLAM) was proposed to improve the accuracy and runtime efficiency of FastSLAM 1.0 and FastSLAM 2.0. This method adopts the landmark measurement with the maximum likelihood, where the particle state is updated before updating the landmark estimate. Also, CESLAM solves the problem of real-time performance. In this paper, a modified version of CESLAM, called adaptive computation SLAM (ACSLAM), as an adaptive SLAM enhances the localization and mapping accuracy along with better runtime performance. In an empirical evaluation in a rich environment, we show that ACSLAM runs about twice as fast as FastSLAM 2.0 and increases the accuracy of the location estimate by a factor of two.

Keywords Fastslam · CESLAM · Particle filter · Extended kalman filter

1 Introduction

Simultaneous localization and mapping (SLAM) is a crucial part of robot navigation in unknown environments. It is necessary, whenever a robot needs to locate itself and build a map of the environment simultaneously [1]. Using the landmark

D.-W. Kung (✉) · C.-C. Hsu · W.-Y. Wang
Department of Electrical Engineering, National Taiwan Normal University,
162 He-Ping East Rd., Sec. 1, Taipei, Taiwan
e-mail: davidkung727@gmail.com

C.-C. Hsu
e-mail: jhsu@ntnu.edu.tw

W.-Y. Wang
e-mail: wywang@ntnu.edu.tw

J. Baltes
Department of Computer Science, University of Manitoba, 77 Chancellor Ave,
R3T 2N2 Winnipeg, MB, Canada
e-mail: jacky@cs.umanitoba.ca

J.-H. Kim et al. (eds.), *Robot Intelligence Technology and Applications 4*,
Advances in Intelligent Systems and Computing 447,
DOI 10.1007/978-3-319-31293-4_7

measurements and odometry of the robot, we can calculate the robot's position and build a map of the environment. There are two popular approaches to SLAM: EKF-SLAM [2] and FastSLAM [3]. The EKF-SLAM is based on extended Kalman filter (EKF), using landmark measurement and odometry information to calculate the EKF's covariance matrix, and then conducts the prediction and correction to finish the localization and mapping. But EKF-SLAM can only converge at a fast speed when the robot system is linear. Several attempts were proposed to address this problem for nonlinear systems. For example, Dissanayake et al. describe a strategy for deleting unsuitable landmarks to improve the accuracy of robot's position and landmark estimation [4]. There are some soft-computing approaches [5–7] that also enhance the EKF-SLAM algorithm. Due to computational complexity, increasing the number of landmarks results in squaring the size of the covariance matrix.

Section 2 presents an overview of related work. Section 3 presents a detailed description of the ACSLAM algorithm. Empirical results of several realistic environments are shown in Sect. 4. The paper concludes with Sect. 5.

2 Related Work

In order to speed up EKF-SLAM, some FastSLAM algorithms are proposed that use Rao-Blackwell normalization on particle filters (RBPF) [8]. The FastSLAM algorithm separates the SLAM problem into localization and mapping phases, where localization uses particle filters (PF) and the mapping uses EKF. Unlike EKF-SLAM, FastSLAM allows each landmark to be calculated by a single EKF, respectively. This can avoid the cost of dealing with large matrices. FastSLAM has two versions, FastSLAM 1.0 [3] and FastSLAM 2.0 [1]. FastSLAM 1.0 only uses odometry to estimate the robot's position, while FastSLAM 2.0 adds the recent sensor measurements to improve estimation accuracy. Though FastSLAM 2.0 seems better than FastSLAM 1.0, the run-time of FastSLAM 2.0 can increase dramatically when the number of landmarks increases.

CESLAM [9] is proposed to solve this problem. Taking advantage of FastSLAM 1.0, and FastSLAM 2.0, CESLAM improves the estimation accuracy and computational efficiency. In the beginning, CESLAM only uses odometry information to estimate the robot's position and updates the particle state and landmark information when a measurement has a maximum likelihood. This paper proposes a modified SLAM algorithm derived from CESLAM, called adaptive computation SLAM (ACSLAM) which improves runtime performance of the algorithm.

3 ACSLAM for Mobile Robots

The number of particles is a crucial parameter in FastSLAM 1.0, FastSLAM 2.0, and CESLAM. Too few particles will result in poor localization accuracy or aliasing problems, but too many particles will result in poor runtime, since the runtime grows linearly with the number of particles.

The proposed ACSLAM generates a probabilistic guess of the robot's position similar to CESLAM. The main difference between the new ACSLAM algorithm and CESLAM is that ACSLAM tries to reduce the runtime of the algorithm by dynamically changing the number of particles.

To improve the runtime, ACSLAM uses the effective sample size (ESS) [10] to decide the next generation's particle number. If the value of ESS is small, it means the difference between each particle's importance weight is large. Then, we will increase the particle number. If the value of ESS is large, it means the differences between each particle's importance weight is small. Next, we decrease the particle number. Figure 1 shows the flowchart of the proposed ACSLAM. Detailed descriptions are listed below:

3.1 *Generate a New Proposal*

For particle $m = 1, \ldots, M$, consider the control input u_t to generate a probabilistic guess of the robot's position

$$x_t^m \sim p\left(x_t | x_{t-1}^m, u_t\right) \tag{1}$$

3.2 *Data Association*

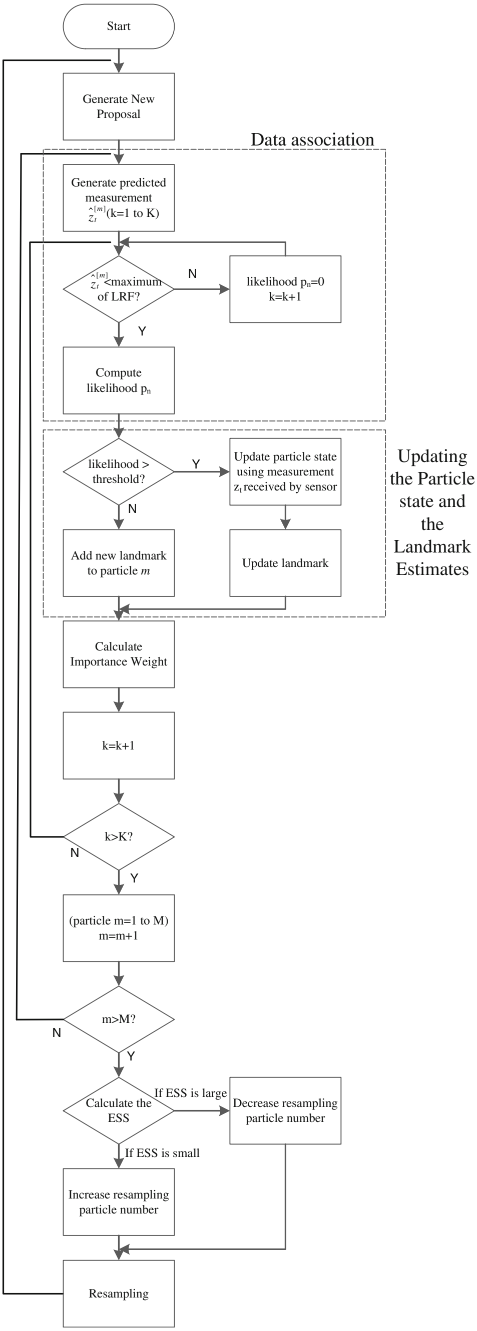
Use the robot's position and the information of landmarks $k = 1 \ldots, N$ in particle m to generate estimated measurement.

$$\hat{z}_t^m = g\left(\hat{x}_t^m, \theta_{k,t-1}^m\right) \tag{2}$$

If the estimated measurement is out of the laser range, then we will set the estimated measurement's likelihood as zero and skip further calculation. Otherwise, we use the estimated measurement to compute the likelihood by using the following equation.

$$p_k = \frac{1}{\sqrt{|2\pi S_k|}} \exp\left[-\frac{1}{2}\left(z_t - \hat{z}_{k,t}\right)^T \left(S_k\right)^{-1} \left(z_t - \hat{z}_{k,t}\right)\right] \tag{3}$$

Fig. 1 Flowchart showing the ACSLAM


Start
Generate New Proposal
Data association
Generate predicted measurement $\hat{z}_t^{[m]}$(k=1 to K)
$\hat{z}_t^{[m]}$<maximum of LRF?
N
likelihood p_n=0 k=k+1
Y
Compute likelihood p_n
likelihood > threshold?
Y
Update particle state using measurement z_t received by sensor
Updating the Particle state and the Landmark Estimates
N
Add new landmark to particle *m*
Update landmark
Calculate Importance Weight
k=k+1
k>K?
N
Y
(particle m=1 to M) m=m+1
m>M?
N
Y
Calculate the ESS
If ESS is large
Decrease resampling particle number
If ESS is small
Increase resampling particle number
Resampling

Where z_t is an actual measurement gathered by a sensor, $\hat{z}_{k,t}$ is the estimated measurement, and S_k is the innovation covariance matrix:

$$\begin{aligned} H_{\theta_k} &= \partial g\left(x_t^m, \theta_{k,t-1}^m\right)/\partial\theta_{k,t-1}^m \\ S_k &= H_{\theta_k}\Sigma_{k,t-1}^m H_{\theta_k}^T + R_t \end{aligned} \tag{4}$$

If any of the estimated measurement's likelihood is larger than the threshold, then the largest one will be used to update the landmark. If not, that means the landmark was not detected before, then we add it as a new landmark to the particle.

3.3 Updating the Particle State and the Landmark Estimates

As CESLAM, if the likelihood is larger than the threshold, this means the landmark has been recorded. Before updating the landmark, we have to use the value to correct the particle's state first. This is to make sure that our particle would not be updated to the wrong position.

$$\begin{aligned} H_{x_t} &= \partial g\left(x_t^m, \theta_{k,t-1}^m\right)/\partial x_t^m \\ \sum\limits_{x_t,t}^{m} &= \left[H_{x_t}^T (S_k)^{-1} H_{x_t} + \left(\sum\limits_{x_t,t}^{m}\right)^{-1}\right]^{-1} \\ u_{x_t,t}^m &= \sum\limits_{x_t,t}^{m} H_{x_t}^T (S_k)^{-1}\left(z_t - \hat{z}_t^m\right) + u_{x_t,t-1}^m \end{aligned} \tag{5}$$

Next, we use the particle's state to calculate the new estimated measurement to update the landmark estimation by EKF.

$$\begin{aligned} K &= \sum\limits_{n_t,t-1}^{m} H_{\theta_{\hat{n}}}^T S_{\hat{n}}^{-1} \\ u_{\hat{n},t}^m &= u_{\hat{n},t-1}^m + K\left(z_t - \hat{z}_t^m\right) \\ \sum\limits_{\hat{n},t}^{m} &= \left(I - KH_{\theta_{\hat{n}}}\right) \sum\limits_{\hat{n},t-1}^{m} \end{aligned} \tag{6}$$

We add the new landmark to the particle if there is no likelihood larger than the threshold.

$$\begin{aligned} N_t^m &= N_{t-1}^m + 1 \\ u_{\hat{n},t}^m &= g^{-1}\left(x_t^m, \hat{z}_{\hat{n},t}\right) \\ \sum\limits_{\hat{n},t}^{m} &= \left(H_{\theta_{\hat{n}}}^T R_t^{-1} H_{\theta_{\hat{n}}}\right)^{-1} \end{aligned} \tag{7}$$

3.4 Calculating Importance Weight and Effective Sample Size

We calculate the importance weight by the following equation:

$$w_t^m = \frac{\text{target distribution}}{\text{proposal distribution}} = \frac{p(x^{t,m}|z^t, u^t, n^t)}{p(x^{t-1,m}|z^{t-1}, u^{t-1}, n^{t-1})p\left(x_t^m|x^{t-1,m}, z^t, u^t, n^t\right)} \tag{8}$$

Then, we use the importance weight to calculate the effective sample size (ESS).

$$ESS = \frac{M}{1 + cv_t^2} \tag{9}$$

M is the particle number, cv_t^2 is the importance weight's coefficient of variation.

$$cv_t^2 = \frac{1}{M}\sum_{i=1}^{M} (Mw(i) - 1)^2 \tag{10}$$

According to the value of ESS, we decrease resampling particle number if ESS is large, and we increase the resampling particle number if ESS is small.

3.5 Resampling

After each particle is assigned an importance weight and we have the next generation's particle number, a tournament selection [11] mechanism is adopted for resampling.

4 Simulation Results

To evaluate the performance, we compare FastSLAM 1.0 and FastSLAM 2.0 with the proposed method. The software environment uses Visual Studio 2010 with the OpenCV library. The map can separate in sparse and compressed block, while the dimension is 500 * 500 pixels. A robot can move 10 pixels or rotate 15° per step. Since real world application of the odometry consists of errors, therefore, the Gaussian noise is introduced with distance and rotation. The simulation of laser range is 180°. The lasers and each landmark in the particle will generate 181 measurements. The maximum distance of the laser range finder is 100 pixels. A robot will need to move 100 steps along the path in each run.

In the proposed method, the maximum particle number we set is 20, and the minimum particle number we set is 10. Once we run the process, the average particle number is 16, then we set the other algorithm's particle number as 16.

Figure 2 depicts the simulation results of the three different SLAM algorithms. The pink points are the true path of the robot trajectory, and gray points are the localization from the best particle. The green circles are the particle, and the yellow circle which is overlapped with the convergent green particles is the robot. The black dots with brown circle are the landmarks, and the red circles are estimations of landmarks. The laser range scanning is shown by blue lines. As the simulation shown in Fig. 2, the robot's localization and landmark estimation in the ACSLAM are more accurate than FastSLAM 1.0 and FastSLAM 2.0. Figure 3 shows the relation between each step and the localization error. Apparently, the compressed block, three algorithms' localization errors are small before counting to the forty fifth step. But after the forty fifth step, the proposed method can adjust the particle number according to the environment as the landmark becomes sparse. As a result, the proposed method outperforms others.

Table 1 shows the localization results of the simulation in different SLAM algorithms. The 'localization error' represents the error between estimated and true robot's position; 'localization RMSE' means the RMSE between estimated and true robot's position. In the table, the performance of ACSLAM in localization is better than FastSLAM 1.0 and FastSLAM 2.0.

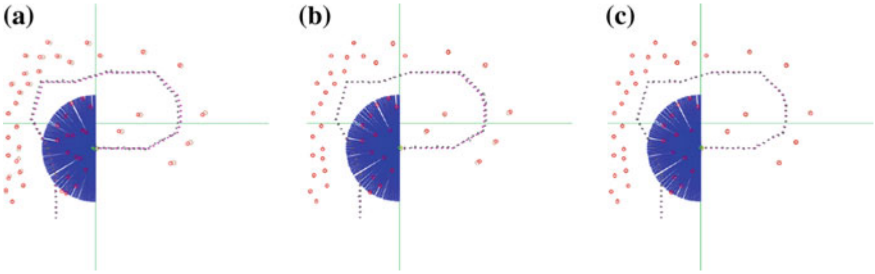


Fig. 2 Simulation results of different SLAM algorithms. **a** FastSLAM 1.0. **b** FastSLAM 2.0. **c** ACSLAM

Fig. 3 Simulation's localization error

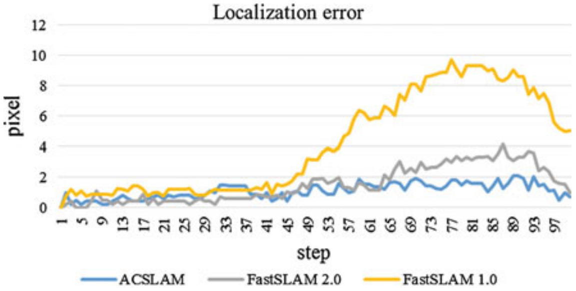


Table 1 Simulation localization results of different SLAM algorithms

SLAM algorithms	Localization error (pixel)	Localization RMSE (pixel)
FastSLAM 1.0	4.17	3.26
FastSLAM 2.0	1.51	1.16
ACSLAM	1.09	0.59

Table 2 Runtime efficiency of different SLAM algorithms

SLAM algorithms	Runtime (s)
FastSLAM 1.0	62.32
FastSLAM 2.0	100.67
ACSLAM	58.84

Table 2 shows the runtime efficiency in different SLAM algorithms. Since the computational time can be reduced, the proposed method performs better than FastSLAM 1.0 and FastSLAM 2.0.

5 Conclusion

In this paper, we propose an extension of CESLAM. The goal is to improve the accuracy, when the particle's measurement has a maximum likelihood in the known landmark. We update the particle state before updating landmark estimation. The proposed algorithm also uses the effective sample size to adjust the particle calculation number so that we can improve the runtime efficiency. As a result, the proposed ACSLAM adopts the advantages of FastSLAM 1.0, FastSLAM 2.0, and CESLAM to ensure better estimation accuracy and runtime in the simulations.

Acknowledge This research is partially supported by the "Aim for the Top University Project" and "Center of Learning Technology for Chinese" of National Taiwan Normal University (NTNU), sponsored by the Ministry of Education, Taiwan, R.O.C. and the "International Research-Intensive Center of Excellence Program" of NTNU and Ministry of Science and Technology, Taiwan, under Grants no. MOST 104-2911-I-003-301 and MOST 103-2221-E-003-027.

References

1. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM 2.0: an improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In: Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI), pp. 1151–1156 (2003)
2. Smith, R.C., Cheeseman, P.: On the representation and estimation of spatial uncertainty. Int. J. Robot. **5**, 56–58 (1986)

3. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM: a factored solution to the simultaneous localization and mapping problem. In: Proceedings of AAAI National Conference on Artificial Intelligence, pp. 593–598 (2002)
4. Dissanayake, G., Williams, S.B., Durrant-Whyte, H., Bailey, T.: Map management for efficient simultaneous localization and mapping (SLAM). Auton. Robots **12**, 267–286 (2002)
5. Chatterjee, A.: Differential evolution tuned fuzzy supervisor adapted, extended Kalman filtering for SLAM problems in mobile robots. Robotica **27**, 411–423 (2009)
6. Chatterjee, A., Matsuno, F.: A neuro-fuzzy assisted extended Kalman based approach for simultaneous localization and mapping (SLAM) problems. IEEE Trans. Fuzzy Syst. **15**, 984–997 (2007)
7. Chatterjee, A., Matsuno, F.: A geese PSO tuned fuzzy supervisor for EKF based solutions of simultaneous localization and mapping (SLAM) problems in mobile robots. Expert Syst. Appl. **37**, 5542–5548 (2010)
8. Murphy, K.: Bayesian map learning in dynamic environments. Neural Inf. Proc. Syst. **12**, 1015–1021 (2000)
9. Yang, C.-K., Hsu, C.-C., Wang, Y.-T.: Computationally efficient algorithm for simultaneous localization and mapping (SLAM). In: Proceedings of the IEEE International Conference on Networking, Sensing and Control (ICNSC), pp. 328–332 (2013)
10. Liu, J.S., et al.: A theoretical framework for sequential importance sampling and resampling. In: Doucet, A., De Freitas, N., Gordon, N.J. (eds.) Sequential Monte Carlo in Practice. Springer (2001)
11. Miller, B. L., Goldberg, D. E.: Genetic Algorithms, Tournament Selection, and the Effects of Noise, IlliGAL Report No. 95006 (1995)