The Knowledge Engineering Review, Vol. 34, e23, 1 of 18.
doi:10.1017/S0269888919000183

Adaptive computational SLAM incorporating strategies of exploration and path planning

JACKY BALTES, DA-WEI KUNG, WEI-YEN WANG and CHEN-CHIEN HSU

Department of Electrical Engineering, National Taiwan Normal University, Taipei, Taiwan
e-mail: jhsu@ntnu.edu.tw

Abstract

Simultaneous localization and mapping (SLAM) is a well-known and fundamental topic for autonomous robot navigation. Existing solutions include the FastSLAM family-based approaches which are based on Rao–Blackwellized particle filter. The FastSLAM methods slow down greatly when the number of landmarks becomes large. Furthermore, the FastSLAM methods use a fixed number of particles, which may result in either not enough algorithms to find a solution in complex domains or too many particles and hence wasted computation for simple domains. These issues result in reduced performance of the FastSLAM algorithms, especially on embedded devices with limited computational capabilities, such as commonly used on mobile robots. To ease the computational burden, this paper proposes a modified version of FastSLAM called Adaptive Computation SLAM (ACSLAM), where particles are predicted only by odometry readings, and are updated only when an expected measurement has a maximum likelihood. As for the states of landmarks, they are also updated by the maximum likelihood. Furthermore, ACSLAM uses the effective sample size (ESS) to adapt the number of particles for the next generation. Experimental results demonstrated that the proposed ACSLAM performed 40% faster than FastSLAM 2.0 and also has higher accuracy.

1 Introduction

Simultaneous localization and mapping (SLAM) is a significant algorithm for robots to navigate in unknown environments. The pose of the robot and landmarks are estimated and built based on odometry as well as expected sensory measurements. Throughout the existing SLAM methods, there are two well-known approaches including extended Kalman filter (EKF)-SLAM (Smith and Cheeseman, 1986) and FastSLAM (Montemerlo *et al.*, 2002). The former one employs odometry and expected sensory measurements to derive the mean and covariance of the EKF, which are accordingly used to estimate a fully correlated posterior over landmarks and robot poses through incremental prediction and correction steps. EKF-SLAM tackles the SLAM problem very well; however, EKF-SLAM converges very fast only when the robot system is linear, which is usually not practical enough in a real world. Hence, in order to speed up the EKF-SLAM under nonlinear system and environment, there are several contributions proposed to improve the overall performances. In Dissanayake *et al.* (2002) and Williams *et al.* (2002), map management methods were proposed to avoid unsuitable landmarks, increasing the accuracy of the robot pose and landmarks estimation. Several soft-computing approaches (Chatterjee and Matsuno, 2007; Chatterjee, 2009; Chatterjee and Matsuno, 2010) were also presented to improve the robustness of EKF-SLAM algorithms. Nevertheless, the computational complexity and sensitivity to failures in data association still remain two significant and serious shortcomings for EKF-SLAM family, which is no exception for the above-mentioned approaches.

To overcome the drawbacks of EKF-SLAM, FastSLAM algorithms were proposed based on a Rao–Blackwellized particle filter (RBPF) (Murphy, 2000). The basic idea of FastSLAM methods is that the SLAM problem is decoupled into localization and mapping problems, where the former one uses particle filters (PFs) to track the pose of the robot, whereas the latter one adopts EKF to converge the pose of landmarks. Different from EKF-SLAM, FastSLAM allows each landmark to be approximated by a corresponding single EKF. By doing so, the heavy computational burden from dealing with large matrix can be avoided, which is the main drawback of the EKF-SLAM. Basically, there are two popular types of FastSLAM including FastSLAM 1.0 (Murphy, 2000) and FastSLAM 2.0 (Montemerlo *et al.*, 2003). FastSLAM 1.0 estimates the pose of the robot only conditioned on the odometry readings, whereas FastSLAM 2.0 introduces sensory observations to improve the prediction of the particle distribution. Nevertheless, the runtime of FastSLAM 2.0 could increase dramatically when the number of landmark increases, especially when the robot is in large-scale environment.

To address the problem of FastSLAM algorithms, several variants (Yang *et al.*, 2013; Hsu and Yang *et al.*, 2017; Hsu and Wang *et al.*, 2017) were proposed to update landmarks only in a certain range with respect to the pose of the robot so that the overall efficiency can be increased, even when the SLAM system performs in large-scale environment (Baltes *et al.*, 2017). However, there is still room for improving the accuracies of localization. As a consequence, this paper aims to propose a SLAM system, namely the 'adaptive computational SLAM' (ACSLAM) based on computationally efficient SLAM (CESLAM) algorithm to increase the overall runtime by adaptively changing the effective sample size (ESS), and provides better estimations of the robot pose as well. Moreover, the ability of exploring the environment autonomously is also employed based on the frontier technique (Yamauchi, 1997; Yamauchi, 1998; Keidar and Kaminka, 2012; Uslu *et al.*, 2015), and potential field method (Goodrich, 2002; Tang *et al.*, 2010) is introduced to provide a path for the robot to move on. Hence, without the need to pre-design paths for the robot, the proposed ACSLAM algorithm is capable of estimating the pose of the robot, building the surrounding environment represented by several landmarks, and exploring the whole indoor environment by moving on a preferable path. To demonstrate the effectiveness of the proposed SLAM system, a laser range finder (LRF) and a Pioneer 3-DX are used to conduct various experiments in indoor environments with lots of obstacles. For easy reference, Figure 1 shows the flow chart of the proposed SLAM system.

The remainder of this paper is as follows: Section 2 describes the preliminaries of FastSLAM algorithms. Section 3 introduces the proposed ACSLAM method. Frontier-based exploration and potential field path planning method are presented in Sections 4 and 5, respectively. Simulations and experimental results are revealed in Section 6 to demonstrate the effectiveness of the proposed SLAM system. Finally, conclusions are drawn in Section 7.

2 Preliminaries of FastSLAM

FastSLAM algorithm is a famous approach to solve the SLAM problem, which separates the SLAM problem into two issues including robot localization and landmarks estimation by using a RBPF described as shown in Equation (1),

$$p\left(s^t, \theta \left| z^t \right., u^t, n^t\right) = p\left(s^t \left| z^t, u^t, n^t\right.\right) \prod_{k=1}^{K} p\left(\theta_k \left| x^t, z^t, u^t, n^t\right.\right) \tag{1}$$

where $s^t = \{s_1, s_2, \ldots s_t\}$ shows the path of the robot from time 0 to t, θ_k $(k = 1, \ldots, K)$ represents the states of landmarks which have been recorded, $z^t = \{z_1, z_2, \ldots, z_t\}$ is the sensory observations obtained from the robot sensor, $u^t = \{u_1, u_2, \ldots u_t\}$ is a set of control inputs from time 0 to t, and $n^t = \{n_1, n_2, \ldots, n_t\}$ shows the number of landmarks being recorded at every single time.

Because the FastSLAM algorithm employs PF to predict the pose of the robot and introduces EKF to estimate the states of landmarks, each state of a particle consists of its own pose as well as the set of mean and covariance of landmarks. This can be described by the following equation,

$$S_t^m = \left\{s_t^m, u_{1,t}^m, \Sigma_{1,t}^m, \ldots, u_{K,t}^m, \Sigma_{K,t}^m\right\} \tag{2}$$

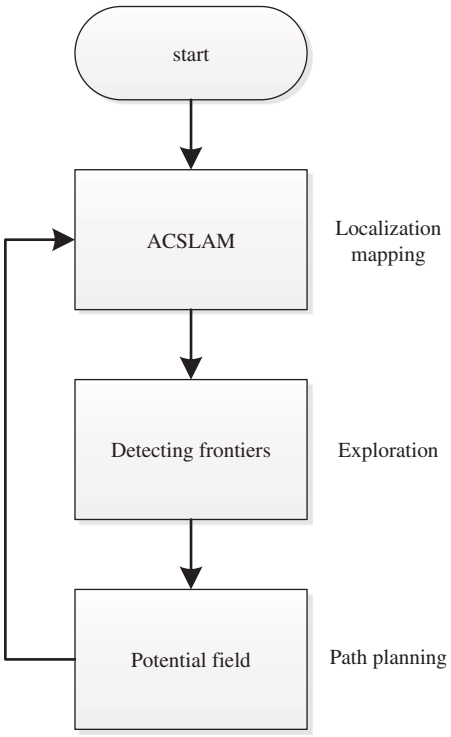

Figure 1 Flow chart of the proposed SLAM system

where s_t^m is the estimation of the state of the robot of the m-th particle at time t, whereas $u_{K,t}^m$ and $\Sigma_{K,t}^m$, respectively, represent the mean and covariance of the k-th landmark. It is worth noting that the state of the robot includes the position s_x and s_y as well as the orientation s_d, that is, $s_t^m = \{s_x, s_y, s_d\}$. Algorithms such as FastSLAM 1.0, FastSLAM 2.0, and CESLAM use fixed size of the particle population during the whole process. However, the overall computational time is actually linearly dependent on the number of particles. In other words, the more particles the system uses, the more computational time is required to perform. On the other hand, less number of particles would lead to inaccurate localization result due to the fact that a sparse probability density function is developed for estimation. As a consequence, an ACSLAM algorithm is proposed in this paper in order to seek for a trade-off number of particles in the process of performing the SLAM algorithm, which is described in detail in the Section 3.

3 Adaptive computational SLAM

To adaptively tune the number of particles, ACSLAM uses the ESS (Uslu *et al.*, 2015) to decide the population size of the next generation. If the value of ESS is large, it implies that the differences of importance weights between particles are relatively small. On the other hand, the similarities between particles are not obvious. Hence, the basic idea of ACSLAM is that it decreases the number of particles if ESS is relatively large, while increases the number of particles if ESS is small enough. Figure 2 shows the complete flow chart of the ACSLAM, and detailed descriptions are listed below as follows.

3.1 Generate a new proposal

For each particle $m = 1 \ldots M$, a probabilistic guess of the pose of the robot is generated according to the previous state and the control input at the current time t, in which the control input is generally regarded as the odometer measurements.

$$\hat{s}_t^m \sim p\left(s_t \left| s_{t-1}^m, u_t\right.\right) \tag{3}$$

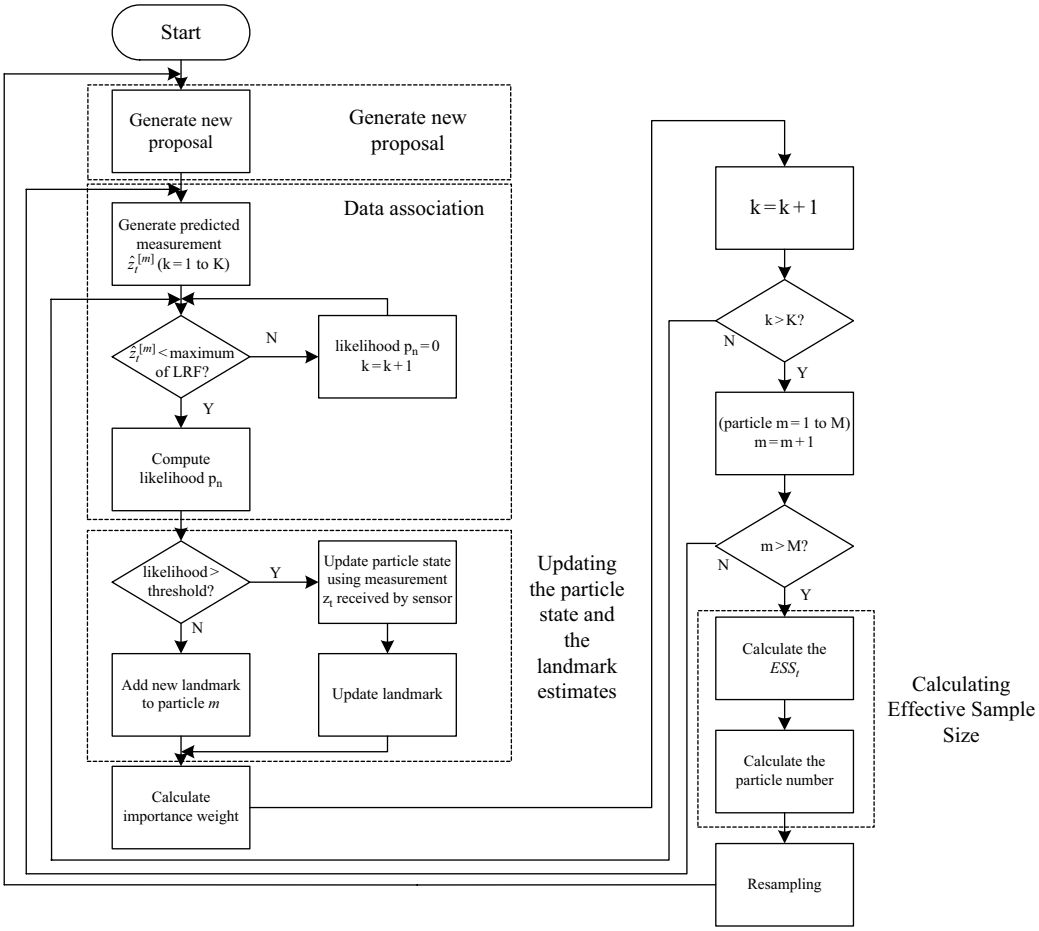

Figure 2 Flow chart of the ACSLAM

3.2 Data association

Estimated measurements are derived according to the sensor model g, where the position of the robot and the state of landmarks are employed.

$$\hat{z}_t^m = g\left(\hat{s}_t^m, \theta_{k,t-1}^m\right) \tag{4}$$

where $k = 1 \ldots N$. To avoid updating unnecessary landmarks, ACSLAM investigates whether the estimated measurement of each corresponding landmark lies inside the range of LRF. If it is outside the range, the corresponding likelihood is assigned to zero and further calculations are neglected. Otherwise, the likelihood of the landmark k is determined based on the estimated sensory measurement $\hat{z}_{k,t}$ derived in Equation (4) and the real sensory observation z_t, as shown in the following equation,

$$p_k^m = \frac{1}{\sqrt{|2\pi V_k|}} \exp\left(-\frac{1}{2}\left(z_t - \hat{z}_{k,t}\right)^T (V_k)^{-1} \left(z_t - \hat{z}_{k,t}\right)\right) \tag{5}$$

In Equation (5), a quadratic form of an innovation covariance matrix V_k for each particle, describing the Jacobian of the expected measurements with respect to the states of landmarks, denoted as H_{θ_k}, with an addictive noise R_t, is shown in the following equation,

$$V_k = H_{\theta_k} \Sigma_{k,t-1}^m H_{\theta_k}^T + R_t \tag{6}$$

where H_{θ_k} is the Jacobian of measurement model with respect to landmark k,

$$H_{\theta_k} = \partial g\left(\hat{s}_t^m, \theta_{k,t-1}^m\right) / \partial \theta_{k,t-1}^m. \tag{7}$$

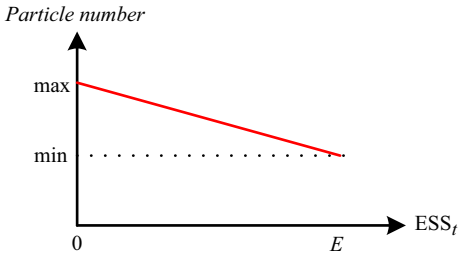

Figure 3 Relationship between the number of particles and ESS

Then, a threshold is given to investigate whether the observed landmarks are required to be created or not. If the likelihood in Equation (5) is larger than the threshold, then the corresponding landmark is regarded as the newly observed one. It is accordingly created as new landmarks for the corresponding particle. In contrast, the corresponding landmark will be updated, which is described in the followings.

3.3 Updating the particle state and the landmark estimates

Before updating the landmark, the state of the particle is prior to be updated, in which the mean and the covariance of the robot pose are determined by Equations (8) *and* (9), respectively,

$$\Sigma_{s_t,t}^{m} = \left[H_{s_t}^{T} (V_k)^{-1} H_{s_t} + \left(\Sigma_{s_t,t-1}^{m} \right)^{-1} \right]^{-1} \tag{8}$$

$$\mu_{s_t,t}^{m} = \Sigma_{s_t,t}^{m} H_{s_t}^{T} (V_k)^{-1} \left(z_t - \hat{z}_t^{m} \right) + \mu_{s_t,t-1}^{m}, \tag{9}$$

where H_{s_t} is the Jacobian of measurement model with respect to the pose of particle m,

$$H_{s_t} = \partial g \left(\hat{s}_t^{m}, \theta_{k,t-1}^{m} \right) / \partial \hat{s}_t^{m}. \tag{10}$$

After updating the pose of particles, landmarks are updated according to the following three equations,

$$K_t = \Sigma_{\hat{n},t-1}^{m} H_{\theta_{\hat{n}}}^{T} V_{\hat{n},t}^{-1} \tag{11}$$

$$\mu_{\hat{n},t}^{k} = \mu_{\hat{n},t-1}^{k} + K_t \left(z_t - \hat{z}_{k,t}^{m} \right) \tag{12}$$

$$\Sigma_{\hat{n},t}^{m} = \left(I - K_t H_{\theta_{\hat{n}}} \right) \Sigma_{\hat{n},t-1}^{m}. \tag{13}$$

3.4 Derivation of effective sample size

The ESS is defined as follows,

$$ESS_t = \frac{M}{1 + cv_t^2}, \tag{14}$$

where M is the number of particles, ${cv_t}^2$ is the coefficient of variation of importance weights,

$$cv_t^2 = \frac{1}{M} \sum_{i=1}^{M} (Mw\,(i) - 1)^2. \tag{15}$$

To adaptively change the particles' population size, the maximum and minimum numbers of particles, $M_{\max}$ and $M_{\min}$, are prior to be pre-designed, respectively, and the value of ESS ranges from zero to E. The relationship between the number of particles and ESS can be illustrated by Figure 3.

As a result, the number of particles of the next-generation P is defined as follows,

$$P = M_{\max} + (ESS_t \times slope). \tag{16}$$

By determining Equation (16), a trade-off population size is provided and the proposed ACSLAM is able to maintain the estimation accuracies in terms of localization while avoiding extra computational time caused by too many particles distributed around the robot pose.

4 Frontier-based exploration

In most SLAM algorithms, control signal is often pre-designated. However, to develop a fully autonomous and intelligent SLAM system, the robot is responsible for exploring the environment by itself. Hence, to solve this problem, this paper employed a strategy based on the frontier-based exploration method (Yamauchi, 1998; Murphy, 2000; Montemerlo, 2003), which provides boundaries, or frontiers, between open spaces and uncharted territories for the robot to move toward. As the robot moves forward to the frontiers, the map is explored and the boundaries are updated as well. As a result, such method accordingly allows the robot to explore the whole unknown environments after no frontiers are remained at all. However, the number of frontiers may increase excessively as the exploration continues. Therefore, to effectively explore the environment, SLAM algorithm incorporating frontier-based method is proposed, which is described in detail below (Figure 4).

4.1 Creations of frontiers

In Figure 5(a), the half circle represents the area covered by the LRF. Let the boundary between area *a* and *b* be denoted as *s*. Then, for the case when obstacles are all located outside the range of sensory observations, five frontiers are set on boundary *s* corresponding to 0, 45, 90, 135, and 180 degrees of laser measurements. These frontiers are represented by the black circles, as shown in Figure 5(b). On the other hand, if the robot encounters an obstacle as shown by a black square in Figure 5(c), the robot only creates four frontiers. All frontiers are stored in the *Frontier curr.* Before the next step, we put the *Frontier curr* into *Frontier temp*, as shown in Figure 6.

4.2 Frontier deletion and the final set

After all frontiers are created, the robot then moves a meter forward and new frontiers are created and stored in the *Frontier curr*. However, we can see from Figure 7 that there are several frontiers created at the previous time being covered by the robot sensor at the current time. These frontiers are shown by the hollow circles in Figure 7 and are not considered as frontiers anymore. Therefore, they are deleted from the *Frontiers temp*. As a result, the *Frontier temp* is updated to include the newly created frontiers stored in *Frontier curr*.

5 Potential field path planning

Since the frontiers imply the directions that the robot has to move toward, the potential field method is employed rather than A* path planning algorithm (Bennewitz *et al.*, 2002) due to the fact that it provides smoother paths and keeps the robot from moving closely along the obstacles, especially for the convex ones. As shown in Figure 8, the whole potential field basically consists of two potentialities according to the goal where the robot has to move to, and the obstacles which are landmarks created by the proposed SLAM system. The two potentialities include the Attractive (*Goal*) and Repulsive (*Obstacle*) one, in which the former allows the robot to move along the directions pointing to the goal and the latter is responsible for keeping the robot from colliding with obstacles.

As shown in Figure 9, let (x_G, y_G) and r_G, respectively, denote the position and the radius of the goal on the field and s_G denotes a threshold describing the distance to the goal. Therefore, the magnitude of the vectors in x and y directions is derived as the following equations,

$$\Delta x_G = \begin{cases} 0, & \text{if } d < r_G \\ \alpha(d - \mathrm{r_G})\cos(\theta), & \text{if } r_G \le d \le s_G + r_G \\ \alpha \mathrm{s_G}\cos(\theta), & \text{if } d > s_G + r_G \end{cases} \tag{17}$$

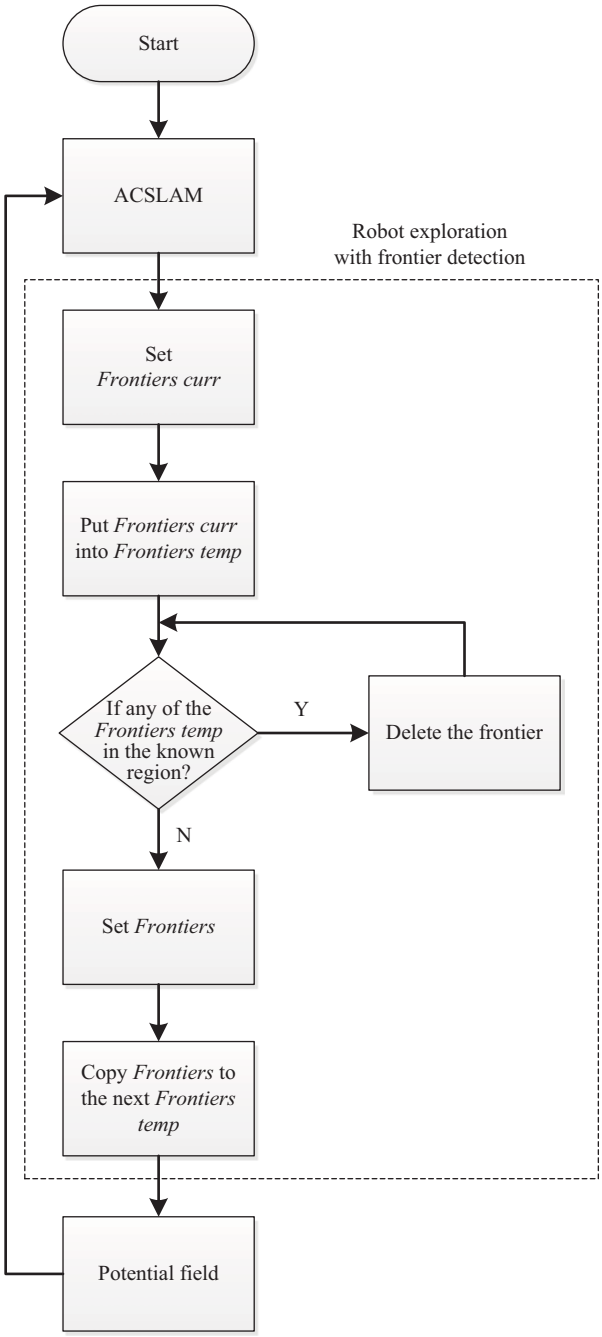

Figure 4 Flow chart of frontier-based exploration

$$\Delta y_G = \begin{cases} 0, & \text{if } d < r_G \\ \alpha(d - \mathrm{r_G})\sin(\theta), & \text{if } r_G \le d \le s_G + r_G \\ \alpha \mathrm{s_G}\sin(\theta), & \text{if } d > s_G + r_G \end{cases} \tag{18}$$

where d and θ are the distance and angles of the goal to the position (x, y). If the robot reaches the goal, Δx and Δy are set as zero since there is no need for the robot to move anymore. On the other hand, if the robot is in the circle with the radius $d = s_G + r_G$, the magnitude of the vector is accordingly set proportional to the distance to the goal. As for the case if the robot locates outside the circle with radius $d = s_G + r_G$, the vector magnitude is set as the maximum value.

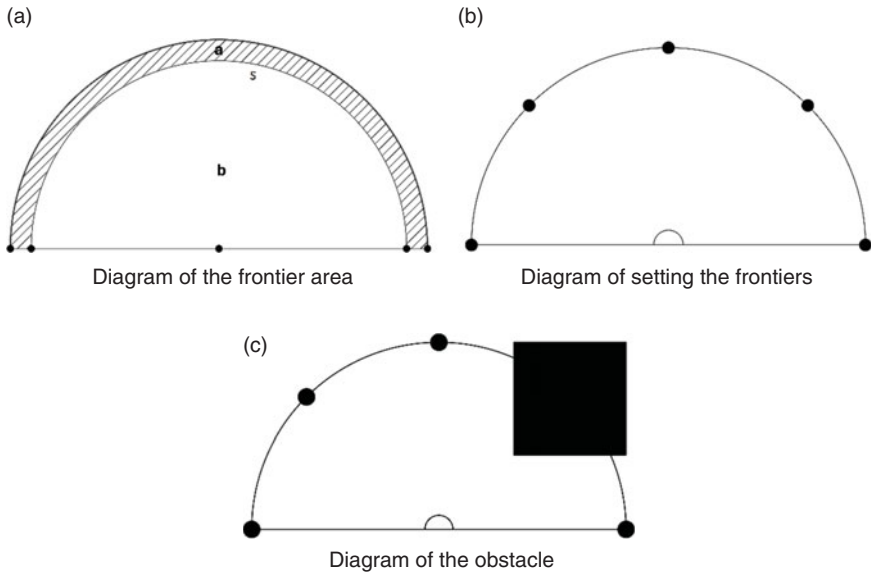

Figure 5 Set the current frontiers from the laser

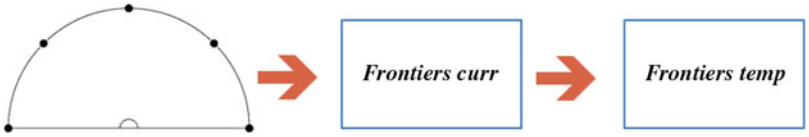

Figure 6 *Frontiers curr* and *Frontiers temp*

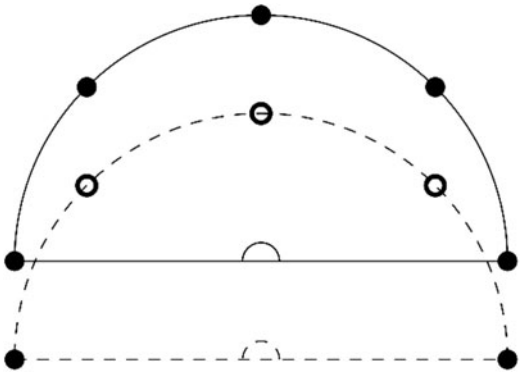

Figure 7 Delete the known frontiers

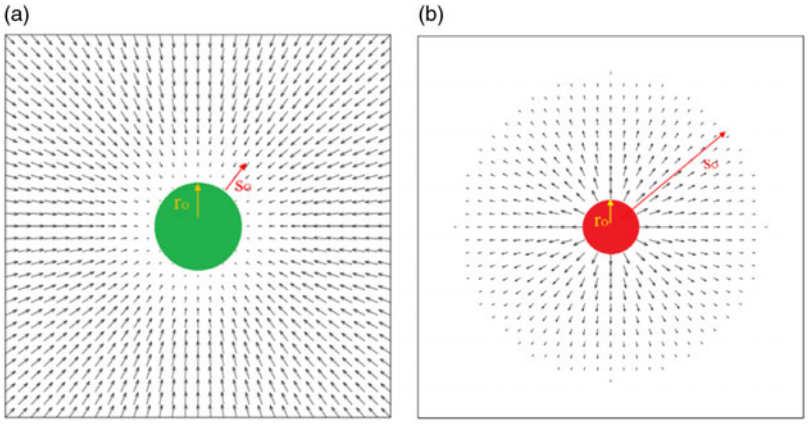

Figure 8 Diagrams of (a) the goal vector and (b) the obstacle vector

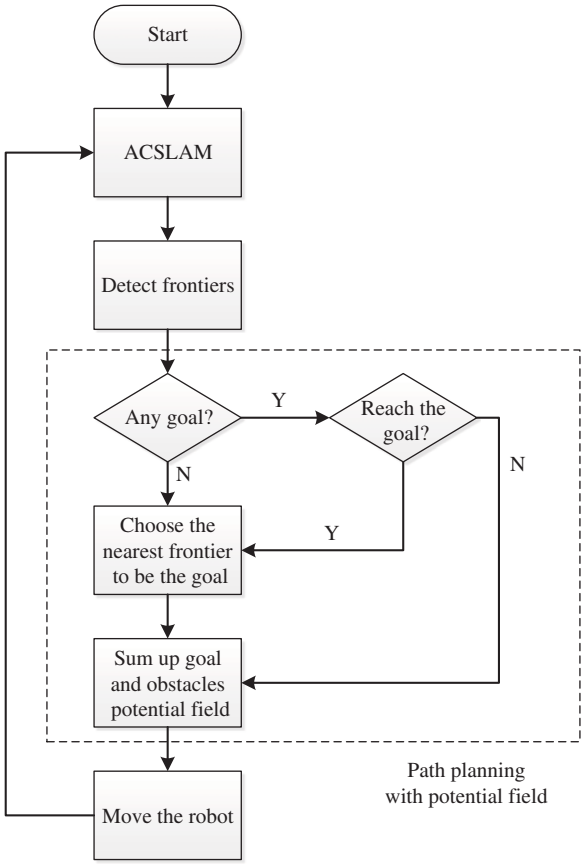

Figure 9 Flow chart of the potential field method incorporated in the proposed SLAM system

Regarding the repulsive potential, let (x_O, y_O) and r_o represent the position and the radius of the obstacle, respectively. The magnitude of the vectors Δx_O^k $(k = 1, ..., K)$ and Δy_O^k $(k = 1, ..., K)$ for the *k-th* obstacle in the map is expressed as follows,

$$\Delta x_O^k = \begin{cases} -sign(\cos(\theta))\infty, & if\ d < r_o \\ -\beta(\mathrm{s_O} + \mathrm{r_O} - d)\cos(\theta), & if\ r_o \le d \le s_o + r_o \\ 0, & if\ d > s_G + r_G \end{cases} \tag{19}$$

$$\Delta y_O^k = \begin{cases} -sign(\sin(\theta))\infty, & if\ d < r_o \\ -\beta(\mathrm{s_O} + \mathrm{r_O} - d)\sin(\theta), & if\ r_o \le d \le s_o + r_o \\ 0, & if\ d > s_G + r_G \end{cases} \tag{20}$$

In the repulsive potential field, vectors are pointed out from the center of the obstacle and the magnitude is infinite. Outside the circle with radius $d = s_O + r_O$, magnitudes are set as zero. Otherwise, the magnitude grows from zero to β (s) if the robot locates outside the circle with radius r_O.

With Δx_G and Δy_G derived by the attractive potential field as well as $\sum_{k=1}^{K} \Delta x_O^k$ and $\sum_{k=1}^{K} \Delta y_O^k$ obtained by the repulsive one, respectively, the integrated potential field for the robot can be determined by using the following equations,

$$\Delta x = \Delta x_G + \sum_{k=1}^{K} \Delta x_O^k \tag{21}$$

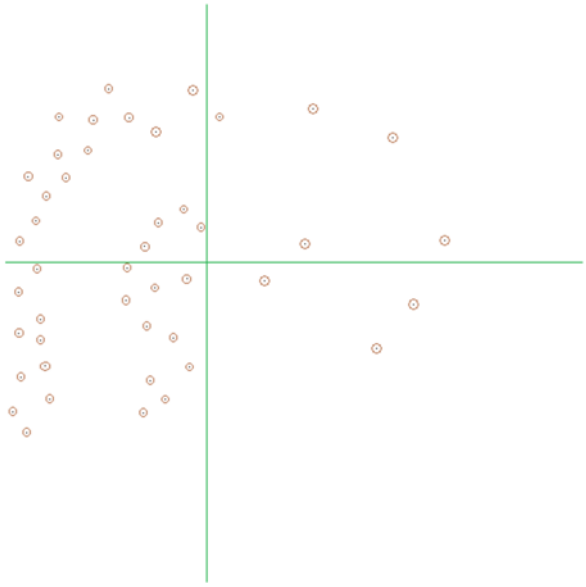

Figure 10 A bit map with resolution of 500 × 500 for simulations

$$\Delta y = \Delta y_G + \sum_{k=1}^{K} \Delta y_O^k \tag{22}$$

where the velocity $v = \sqrt{\left(\Delta x^2 + \Delta y^2\right)}$ as well as the angle $\theta = \tan^{-1}\left(\Delta y / \Delta x\right)$ can be derived for the robot. The goal for the robot to move toward is designed as the nearest frontier in the *Frontiers*. From the aforementioned, the proposed system is now capable of exploring the environment by moving on the path provided by the potential field method after frontiers are obtained. A brief overview of the potential field algorithm incorporated in the proposed SLAM system can be seen by the flow chart shown below.

6 Simulations and experiments

6.1 Experimental setup

To evaluate the performances of the proposed SLAM system, several simulations and experiments are conducted. The software environment is Visual Studio 2010 Express with OpenCV library, and a personal computer with Intel Core i7 3.06 GHz, 6 GB RAM is employed for simulations. On the other hand, a laptop with Intel Core i5-3317U 1.70 GHz, 6 GB RAM is used as the processing platform and a SICK LMS100 LRF and a Pioneer 3-DX mobile robot are provided to obtain sensory measurements and odometer readings.

There are two cases designed for conducting simulations and experiments, where a bit map with a resolution of 500 × 500 pixels is introduced for simulations, and an indoor environment with the size of 480 cm × 480 cm is provided for experiments. Parameters such as $M_{\max}$ and $M_{\min}$ mentioned in Figure 3, respectively, are set as 20 and 10. It is worth noting that there is no Gaussian noise employed in the second case of simulations, while other cases contain zero mean Gaussian noises with a variance of 0.1 pixel.

6.2 Simulation results

6.2.1 Case I

The map provided in this case is shown in Figure 10, while the dynamic number of particles for each step is presented in Figure 11. The average population size in the proposed ACSLAM is 16; hence, 16 particles are used for other SLAM algorithms to reach a fair comparison.

Figure 12 shows the simulation results of different algorithms, where the path in gray is the localization results of the best particle and the path in pink is the ground truth of the robot trajectory. Green dots

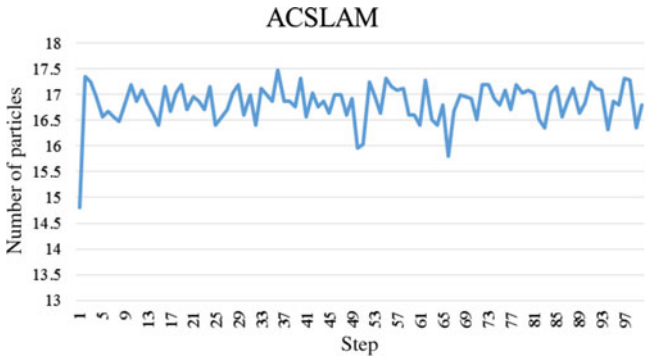

Figure 11 Dynamic number of particles

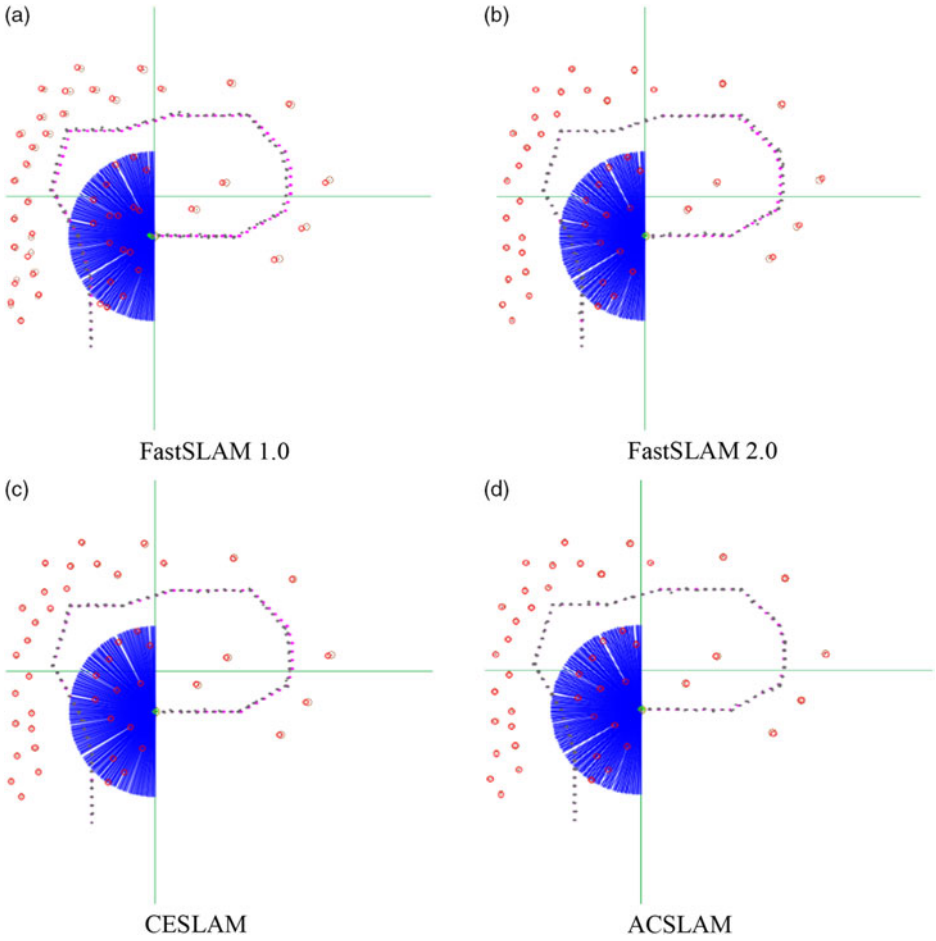

Figure 12 Simulations of various SLAM algorithms

represent all particles in the SLAM algorithm. The brown circles with a black dot are the landmarks. The estimations of landmarks are shown in red circles. LRF is shown by blue lines. As demonstrated in Figure 12, the accuracy of robot localization proposed by ACSLAM is better than FastSLAM 1.0, FastSLAM 2.0 and CESLAM.

In Figure 13, the blue line represents the mean (m) of the x localization error for each step, while the black lines are the covariance (v) of the x localization error for each step. As for the y localization error, Figure 14 shows the simulation results. Table 1 shows the average localization error and runtime results of 10 independent runs for various SLAM algorithms. The average localization error represents the average localization error between the estimated pose of robot and the real ones.

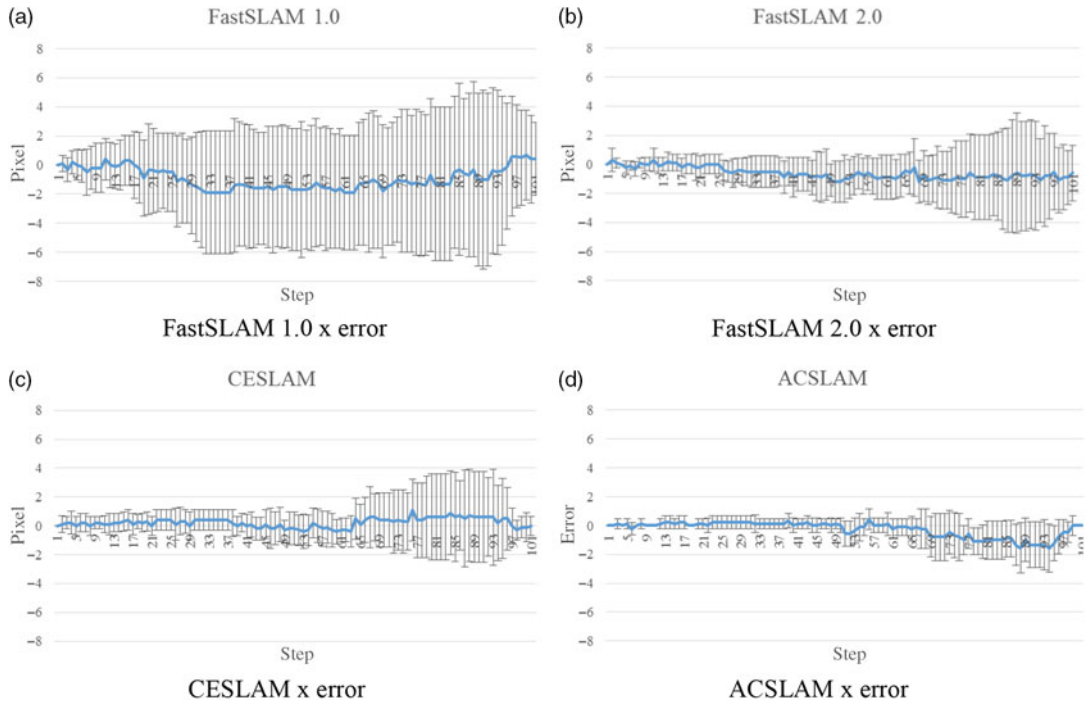

Figure 13 Mean and covariance of x error

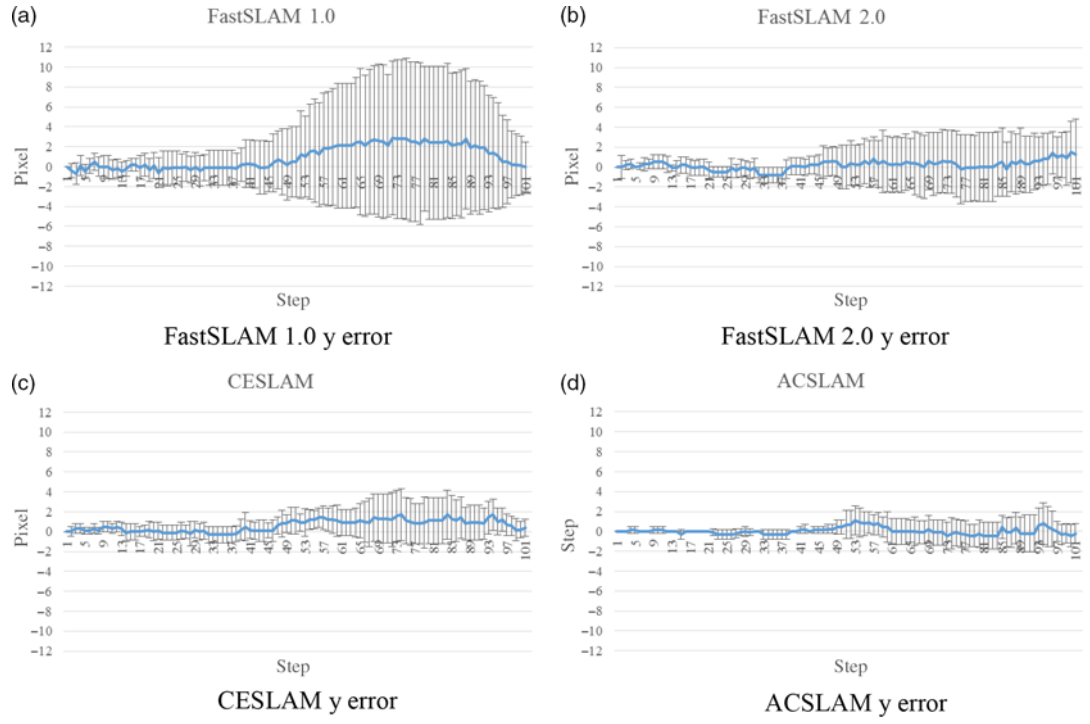

Figure 14 Mean and covariance of y error

6.2.2 Case II

We use two maps to challenge the proposed ACSLAM as shown in Figure 15 (a) and (b), where the former one consists of sparse landmarks and the latter is provided with several compressed landmarks. The LRF used in our simulations ranges from 0 to 180 degrees, and the maximum distance of the LRF is 110 pixels.

Table 1 Simulation results

SLAM algorithms	FastSLAM 1.0	FastSLAM 2.0	CESLAM	ACSLAM
Runtime (s)	67.26	99.46	56.53	56.64
Average localization error (pixel)	6.42	2.15	1.95	1.15

SLAM, simultaneous localization and mapping.

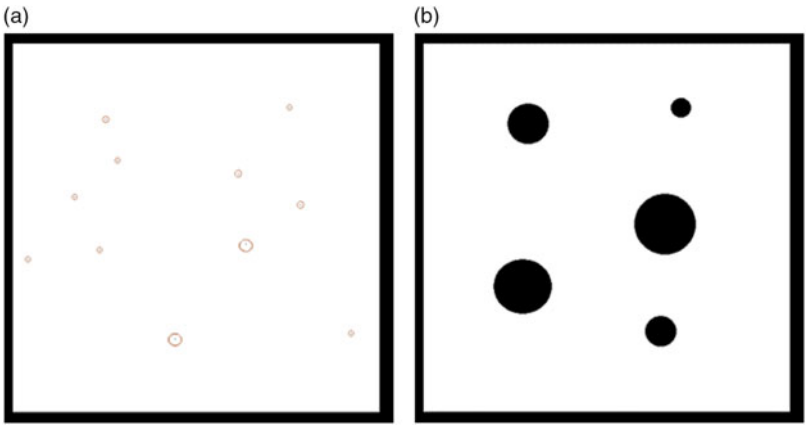

Figure 15 Two different maps for the simulation

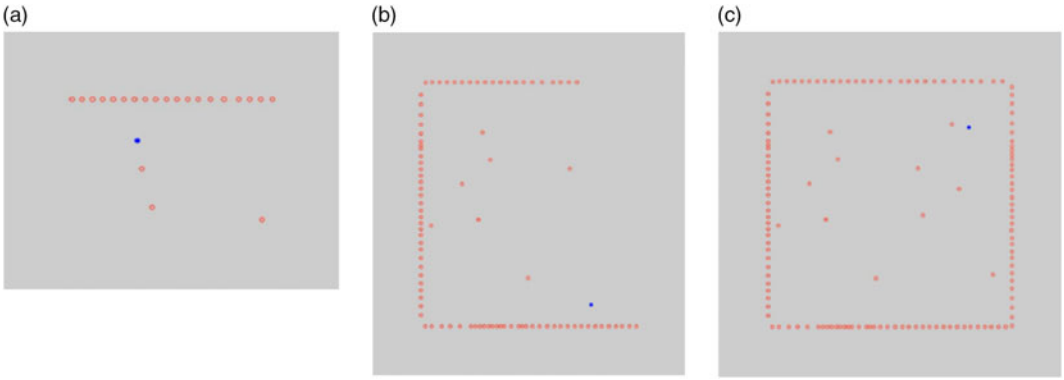

Figure 16 The process of ACSLAM for mapping

Figure 16 depicts the simulation results of the ACSLAM processes. The blue dot is the position of the robot, and the red circles are the estimations of landmarks. We can see from the figure that robot successfully explores the whole environment based on the proposed SLAM system.

Figure 17 shows the processes of setting the frontiers, which is in correspondence with Figure 16. The blue dot is the pose of the robot, while the green dot is the goal chosen from the nearest frontier. The set of *Frontiers* is shown in red dots.

Figure 18 is the potential field shown in correspondence with Figures 16 and 17. The blue lines are laser range scanning. Red circles are estimations of landmarks, and the black lines are the magnitude and direction of the potential field.

As for the map shown in Figure 15 (b), Figure 19 presents the simulation results of the ACSLAM. Identically, the blue dot is the position of robot, and the red circles are estimations of landmarks. Figure 20 shows the frontiers in correspondence with Figure 19.

Figure 21 is the potential field in correspondence with Figures 19 and 20. Table 2 shows the runtime efficiency by using the maps, as shown in Figure 15.

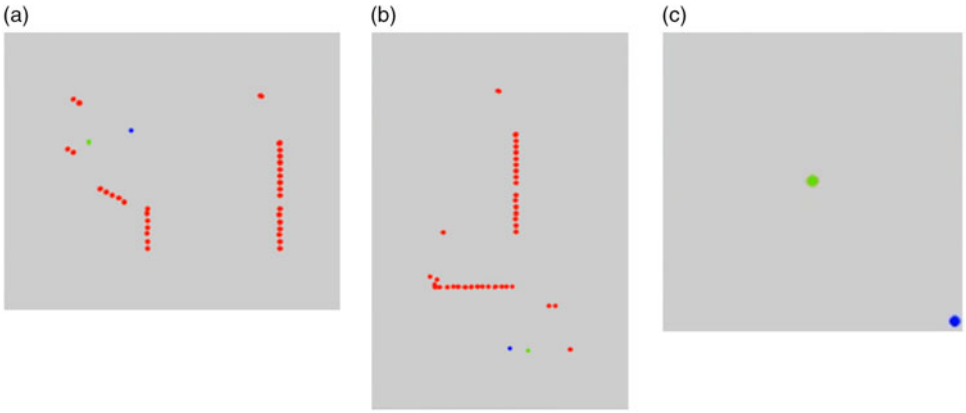

Figure 17 The process of setting the frontiers

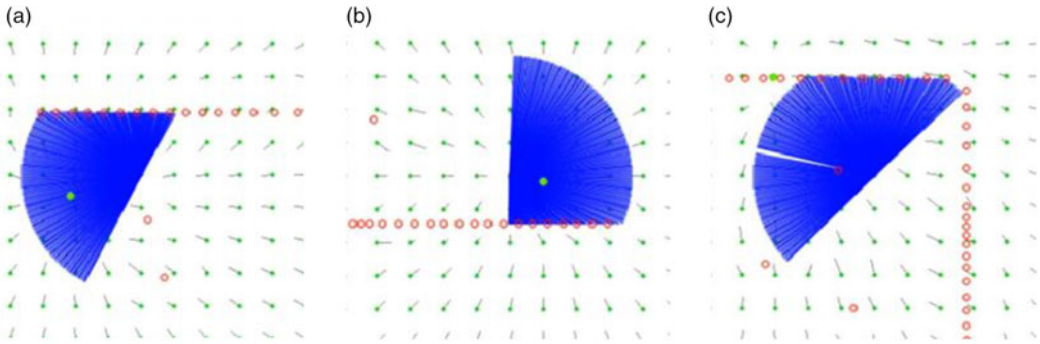

Figure 18 The potential field nearing the robot

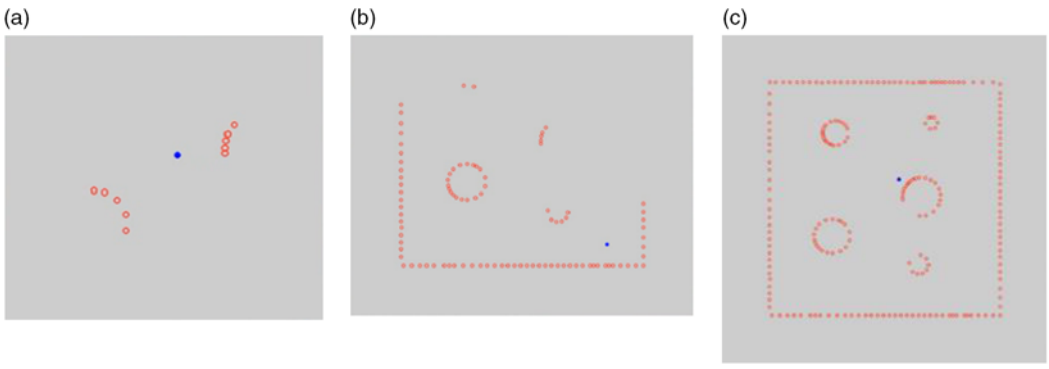

Figure 19 The process of ACSLAM for mapping

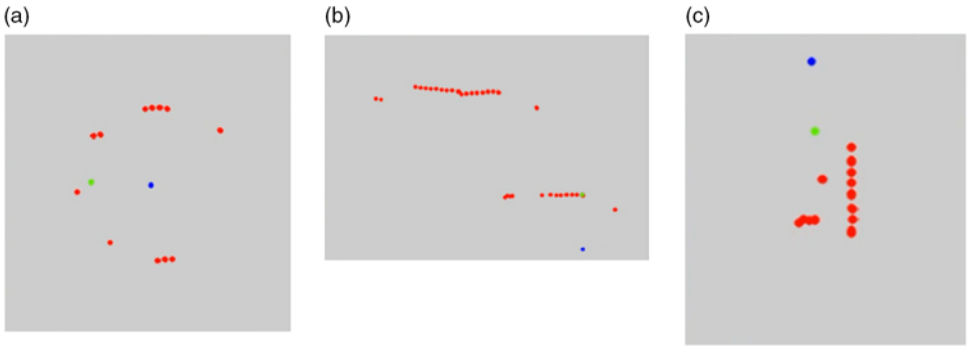

Figure 20 The process of setting the frontiers

Table 2 Runtime efficiency of the exploration adaptive computational simultaneous localization and mapping with different maps

Map	Runtime (s)
Sparse landmarks	854
Compressed landmark	1577

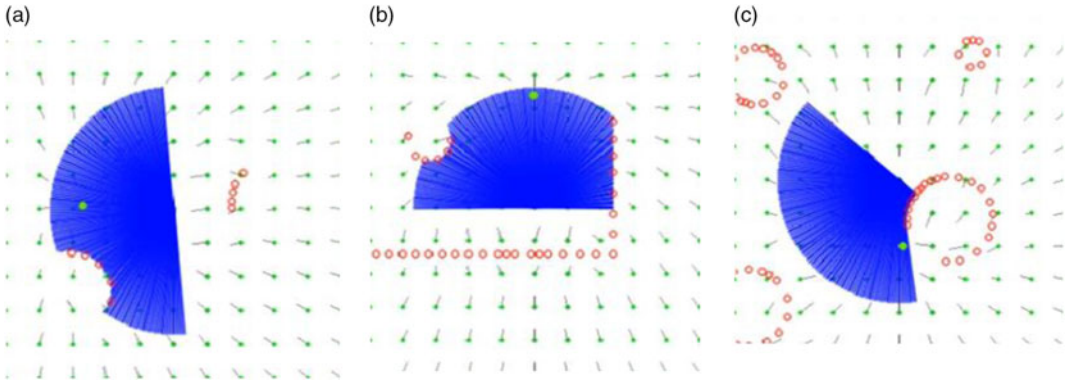

Figure 21 The potential field nearing the robot

Figure 22 Experimental environment of case I

6.3 Experimental results

6.3.1 Case I

In case I, we use the experimental map as shown in Figure 22, to execute the four independent experiments, and the results are shown in Figure 23. Because the robot explores the environment automatically with different initial poses, the paths are totally different for the four experiments. Table 3 shows the average number of particles and the runtime corresponding to Figure 23.

6.3.2 Case II

In case II, using more complicated map to execute the algorithm as shown in Figure 24, four independent experiments are also conducted as well. The experimental results are shown in Figure 25. The path in gray is the localization of the best particle. Table 4 shows the average number of particles and the runtime efficiency. According to the experimental results of the two cases, we can firmly state that the proposed SLAM system is capable of building the map and exploring the environment successfully.

Table 3 Experimental results in case I

ACSLAM	Average number of particles	Runtime (s)
a	17	2340.63
b	17	1200.36
c	17	1620.20
d	17	1201.44

ACSLAM, adaptive computational simultaneous localization and mapping.

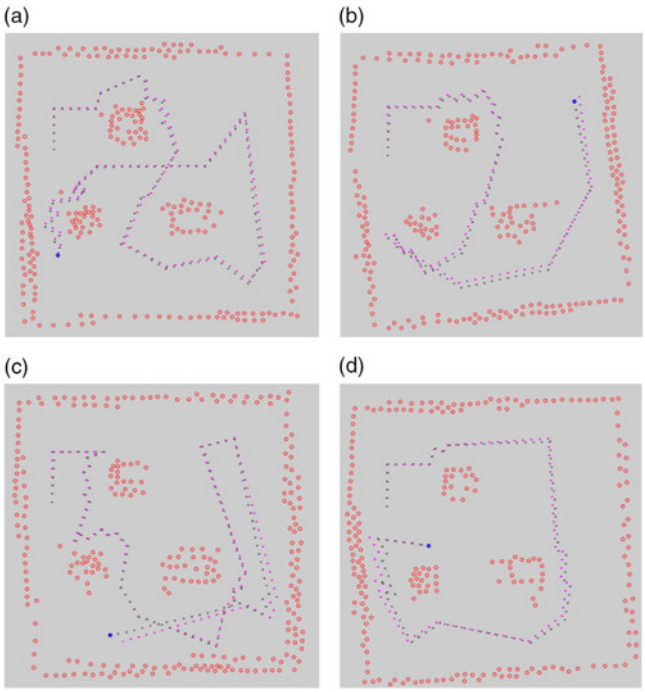

Figure 23 Experimental results in case I

Figure 24 Experimental map in case II

Table 4 Experimental results in case II

ACSLAM	Average number of particles	Runtime (s)
a	17	2220.60
b	17	1681.26
c	17	2034.32
d	18	2009.91

ACSLAM, adaptive computational simultaneous localization and mapping.

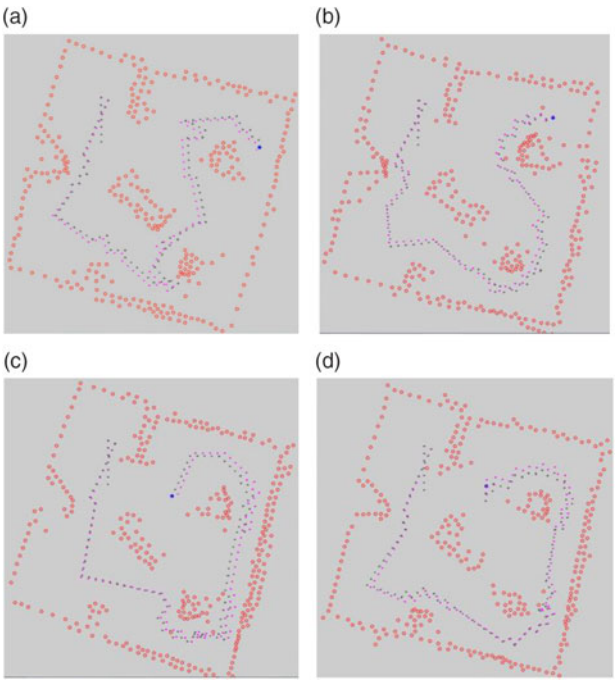

Figure 25 Experimental results in case II

7 Conclusion

In this paper, a SLAM system with the ability of exploring the whole environment while performing the SLAM algorithm concurrently is proposed based on the CESLAM algorithm. By determining the values of ESS to decide the population size of the next generation, the number of particles of ACSLAM can adaptively change as the iteration continues. This not only decreases the overall runtime efficiencies but also increases the accuracies of robot localization. Moreover, in order to explore the environment by the robot itself without the need to predesign a path priorly, a frontier-based exploration method is incorporated to the ACSLAM algorithm. To move to the frontiers, a path is provided by using the potential field method. Several simulations and experiments have verified the effectiveness of the proposed SLAM system in comparison with various other SLAM techniques.

Acknowledgements

This work was financially supported by the 'Chinese Language and Technology Center' of National Taiwan Normal University (NTNU) from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan, and Ministry of Science and Technology, Taiwan, under Grants No. MOST 108-2634-F-003-002,

MOST 108-2634-F-003-003, and MOST 108-2634-F-003-004 (administered through Pervasive Artificial Intelligence Research (PAIR) Labs), as well as MOST 107-2811-E-003-503. We are grateful to the National Center for High-performance Computing for computer time and facilities to conduct this research.

References

Baltes, J., Tu, K. Y., Sadeghnejad S. & Anderson, J. 2017. HuroCup: competition for multi-event humanoid robot athletes. *Knowledge Engineering Review* **32**, e1.

Bennewitz, M., Burgard, W. & Thrun, S. 2002. Finding and optimizing solvable priority schemes for decoupled path planning techniques for teams of mobile robots. *Robotics and Autonomous Systems* **41**, 89–99.

Chatterjee, A. 2009. Differential evolution tuned fuzzy supervisor adapted, extended Kalman filtering for SLAM problems in mobile robots. *Journal of Robotica* **27**, 411–423.

Chatterjee, A. & Matsuno, F. 2007. A neuro-fuzzy assisted extended Kalman filter based approach for simultaneous localization and mapping (SLAM) problems. *IEEE Trans. on Fuzzy Systems* **15**, 984–997.

Chatterjee, A. & Matsuno, F. 2010. A Geese PSO tuned fuzzy supervisor for EKF based solutions of simultaneous localization and mapping (SLAM) problems in mobile robots. *Expert Systems with Applications* **37**, 5542–5548.

Dissanayake, G., Williams, S. B., Durrant-Whyte, H. F. & Bailey, T. 2002. Map management for efficient simultaneous localization and mapping (SLAM). *Journal of Autonomous Robots* **12**, 267–286.

Goodrich, M. A. 2002. Potential fields tutorial. https://phoenix.goucher.edu/~jillz/cs325_robotics/goodrich_potential_fields.pdf

Hsu, C. C., Wang, W. Y., Lin, T. Y., Wang, Y. T. & Huang, T. W. 2017. Enhanced Simultaneous Localization and Mapping (ESLAM) for Mobile Robots. *International Journal of Humanoid Robotics,* **14**, 1750007-1-17.

Hsu, C. C., Yang, C. K., Wang, Y. T., Wang, W. Y. & Chien, C. H. 2017. Computationally efficient algorithm for vision-based simultaneous localization and mapping of mobile robots. *Engineering Computations* **34**, 1217–1239.

Keidar, M. & Kaminka, G. 2012. Robot exploration with fast frontier detection: theory and experiments. In *Proc. of AAMAS*, Valencia, 113–120.

Montemerlo, M., Thrun, S., Koller, D. & Wegbreit, B. 2002. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of AAAI National Conference on Artificial Intelligence*, 593–598.

Montemerlo, M., Thrun, S., Koller, D. & Wegbreit, B. 2003. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proc. of the 16th Int. Joint Conf. on Artificial Intelligence*, 1151–1156.

Murphy, K. 2000. Bayesian map learning in dynamic environments. *Neural Information Proceedings System* **12**, 1015–1021.

Smith, R. C. & Cheeseman, P. 1986. On the representation and estimation of spatial uncertainty. *International Journal of Robotics* **5**, 56–58.

Tang, L., Dian, S., Gu, G., Zhou, K., Wang, S. & Feng, X. 2010. A novel potential field method for obstacle avoidance and path planning of mobile robot. In *Proc. of ICCSIT*, Chengdu, 633–637.

Uslu, E., Cakmak, F., Balcilar, M., Akinci, A., Amasyali, M. F. & Yavuz, S. 2015. Implementation of frontier-based exploration algorithm for an autonomous robot. In *Proc. of INISTA*, Madrid, 1–7.

Williams, S. B., Dissanayake, G. & Durrant-Whyte, H. F. 2002. An efficient approach to the simultaneous localization and mapping problem. In *Proc. of the IEEE International Conference on Robotics and Automation*, Washington, USA, 406–411.

Yamauchi, B. 1997. A frontier-based approach for Autonomous exploration. In *Proceedings of the 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Monterey, CA, 146–151.

Yamauchi, B. 1998. Frontier-based exploration using multiple robot. *Agents* **98**, 47–53.

Yang, C. K., Hsu, C. C. & Wang, Y. T. 2013. Computationally efficient algorithm for simultaneous localization and mapping (SLAM). In *Proc. of the IEEE Int. Conf. Networking, Sensing and Control*, 328–332.