SUGISAKA MASANORI

Journal of Robotics, Networking and Artificial Life
Vol. 9(2); September (2022), pp. 177–182
ISSN (Online):2352-6386; ISSN (Print):2405-9021
https://alife-robotics.org/jrnal.html

Research Article

Online SLAM for Forestry Robot

Sylvain Geiser[1], Sakmongkon Chumkamon[2], Ayumu Tominaga[3], Takumi Tomokawa[2], Eiji Hayashi[2]

[1]*Department of Interdisciplinary Informatics, Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, 680-4, Kawazu, Iizuka-City, Fukuoka, 820-8502, Japan*

[2]*Department of Mechanical Information Science and Technology, Kyushu Institute of Technology, 680-4, Kawazu, Iizuka-City, Fukuoka, 820-8502, Japan*

[3]*National Institute of Technology, Kitakyushu College, 5-20-1 Shii, Kokuraminamiku, Kitakyushu, Fukuoka, 802-0985 Japan*

ARTICLE INFO

Article History

Received 24 November 2021
Accepted 31 May 2022

Keywords

Field robot
Online SLAM
Forestry
Particle filter

ABSTRACT

Less and less Japanese are working in the forestry sector. Due to this lack of workforce, dangerous tasks could be delegated to robots. In order to perform this mission, the latter need embedded localization and mapping systems. This paper discusses the implementation of a particle filter based SLAM solver on a dedicated mobile robot called SOMA, developed at Hayashi Laboratory. The proposed approach has been evaluated in a realistic forest simulation and the first results suggest that it could be used in real time

1. Introduction

In Japan, owing to the combination of aging workers and unattractivity, the forestry sector is facing a significant decrease in terms of workforce, while the demand is conversely increasing.[1,2] In this tense context, mobile robots have been proposed to replace lacking humans, especially for hazardous jobs. The Hayashi Laboratory is building a dedicated prototype for this purpose. This robot, called SOMA,[3] is based on the platform of an All-Terrain Vehicle (ATV). Many additional sensors including odometers, GPS (Global Positioning System), IMU (Inertial Measurement Unit), RGB-D cameras and lidar have been attached to the structure.

In mobile robotics, navigation is an essential topic to deal with, and field robots are not exceptions to the rule. For a long time, the several problems to solve in this context have been unified in the framework of Simultaneous Localization and Mapping (SLAM), but current applications mostly treat separately mapping and localization. The first one is performed offline and once for all, whereas the second one is carried out online. This method tends to be very efficient when the environment where the robot evolves remains unchanged. However, when it is frequently changing, this approach becomes inappropriate and online mapping, worthwhile. This is the reason why, forests being constantly transformed by exploitation, online SLAM is proposed in this article as the problematic to address for this specific application.

Numerous algorithms have been developed in this domain, but FastSLAM has the advantage of being able of handling multimodal beliefs, owing to its logic based on particle filters.[4] This is the main reason why FastSLAM was selected to be implemented on SOMA.

This article is organized in three parts. The first one explains the realized implementation and the undertaken choices. Then, the simulation experiments and parameters are described. Finally, the results are presented and an analysis of them is performed, leading to the conclusion which reviews achievements and remaining goals to reach.

Corresponding author's E-mail: geiser.nathan-sylvain778@mail.kyutech.jp, m-san@mmcs.mse.kyutech.ac.jp, tominaga@kct.ac.jp, tomokawa.takumi163@mail.kyutech.jp, haya@.mse.kyutech.ac.jp, http://www.kyutech.ac.jp/

2. FastSLAM for Forestry Robot

The core of the FastSLAM algorithm is a Rao-Blackwellized particle filter and the environment is described with a feature map. The latter consists of the list of the locations of easy distinguishable elements called features or landmarks. In order to estimate the pose of the robot and the map at the same time, the particles used include, apart from a weight, a pose and a collection of Kalman filters. The latter store the position and associated uncertainty of features.

Each time new sensory data becomes available, two steps are performed: prediction and correction, which are common to every classical Bayes filter used in robotics. During prediction step, the poses of the particles are changed by applying the captured motion of the real robot, corrupted with the modelled motion noise. Then, two different updates are applied during correction step. First, according to received observation and associated noise, the Kalman filters corresponding to features in the sensor visibility scope are updated for each particle and new filters are initialized for previously unseen landmarks, using Kalman equations. In addition, the weights of particles are updated in accordance with observation likelihood.

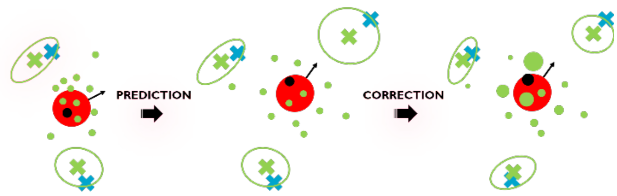

Fig. 1. Steps of Fast SLAM algorithm

The Figure 1 is illustrating this process. In this diagram, the real robot is represented by a red disk indicating its real position and a black line in front of it revealing its heading. Concerning the particles, they are depicted by green points with variable size, located at the position they contain. The bigger a particle, the higher its weight, the more probable it is to embody the true pose of the robot and the true map of the environment. The particle with the highest weight before the new iteration of the algorithm is highlighted in black. Finally, the blue crosses represent the real features, whereas the green crosses are the landmarks in the map of the most probable particle. The green circles around green crosses represent the uncertainty over the positions of features.

Since this algorithm can be used in different situations, our implementation has to adjust to the forest environment particularities. One major parameter to set is the nature of landmarks. Here, the choice of trees seems obvious, due to their concentration and their easily identifiable shape. Moreover, specific motion and observation models also have to be chosen. Their description is the topic of the following sections.

2.1. *Motion*

The robot motion is captured through the odometry delivered by wheels encoders and the selected model breaks the movement in three steps: a first rotation, then a translation, and finally a second rotation. Each of them is supposed to be corrupted with linear gaussian noise (Eq. 1, 2 and 3).

$$\delta rot1 \sim \mathcal{N}\big(\underbrace{\overline{\delta rot1}}_{\mu_{rot1}}, \underbrace{a_{rot}\cdot|\overline{\delta trans}| + b_{rot}\cdot|\overline{\delta rot1}| + c_{rot}}_{\sigma_{rot1}}\big) \tag{1}$$

$$\delta trans \sim \mathcal{N}\big(\underbrace{\overline{\delta trans}}_{\mu_{trans}}, \underbrace{a_{trans}\cdot|\overline{\delta trans}| + b_{trans}\cdot(|\overline{\delta rot1}| + |\overline{\delta rot2}|) + c_{trans}}_{\sigma_{trans}}\big) \tag{2}$$

$$\delta rot2 \sim \mathcal{N}\big(\underbrace{\overline{\delta rot2}}_{\mu_{rot2}}, \underbrace{a_{rot}\cdot|\overline{\delta trans}| + b_{rot}\cdot|\overline{\delta rot2}| + c_{rot}}_{\sigma_{rot2}}\big) \tag{3}$$

where $\delta rot1$, $\delta trans$ and $\delta rot2$ are noisy values of first rotation, translation and second rotation respectively, $\overline{\delta rot1}$, $\overline{\delta trans}$ and $\overline{\delta rot2}$ are corresponding noiseless values, and a_{rot}, b_{rot}, c_{rot}, a_{trans}, b_{trans} and c_{trans} are constants.

2.2. *Observation*

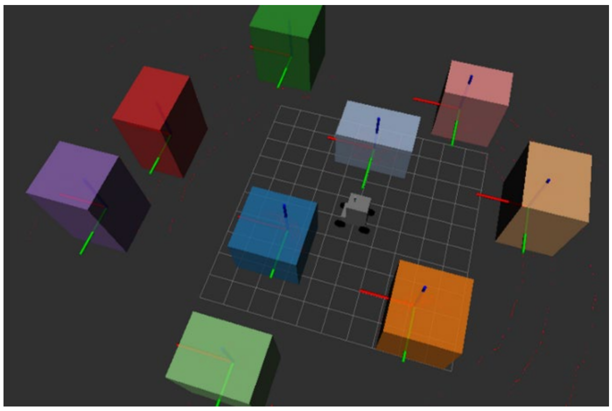

Fig. 2. Clusters made from lidar pointcloud

Concerning observation, the range-bearing model was taken on, because the sensor uses to perceive the environment around the robot is a lidar. Consequently, each individual observation is composed of the distance and the azimuth between the feature and the robot, which defines coordinates in the moving polar reference frame attached to the robot. In order to get them for each tree in the visibility scope, the raw pointcloud has to be processed. First, the ground and the canopy are removed by clipping it in height. Then, 3D euclidian clustering is performed to locate trunks, turning their representation from discrete and noisy points into simple parallelepipeds (Fig. 2). Finally, the centers of gravity of the clusters are taken as the positions of the trees. The returned length of each laser ray is assumed to include linear gaussian noise. Therefore, given some approximations, the extracted coordinates of the trees follow the same noisy distribution (Eq. 4 and 5).

$$d \sim \mathcal{N}(\underbrace{\bar{d}}_{\mu_d}, \underbrace{a_d \cdot |\bar{d}| + b_d}_{\sigma_d}) \tag{4}$$

$$\varphi \sim \mathcal{N}(\underbrace{\bar{\varphi}}_{\mu_\varphi}, \underbrace{a_\varphi \cdot |\bar{d}| + b_\varphi}_{\sigma_\varphi}) \tag{5}$$

where d and φ are the noisy values of distance and azimuth respectively, $\bar{d}$ and $\bar{\varphi}$ are the corresponding noiseless values, and a_d, b_d, a_φ and b_φ are constants.

2.3. *Correspondences and map management*

The correspondence problem, which consists of associating each observation in reality with a feature in the map, can be tackled in various ways. The maximum likelihood approach has been chosen, as it is often used and has proven its efficiency. In FastSLAM, each particle having its own correspondences, a large set of different data associations cohabit throughout the experiment. This diversity tends to lead to better results.

2.3.1. Multiple observations

A tree is rarely alone in the lidar visibility scope. Hence, since several trees are seen at the same time, observations are called multiple. Instead of decomposing these multiple observations into individual ones and executing successive updates for each of them, they are treated as a whole, by using a slightly modified Gale-Shapley algorithm.[5] This choice prevents a common issue encountered using the usual splitting technique: wrong fusion of landmarks. Indeed, in this case, distinct features in reality can be linked to the same feature in the map.

For each observed tree, the likelihood toward each feature in the visibility scope is computed and ordered in a list. At the beginning, the first-ranked landmarks are associated with the corresponding tree. Then, if multiple observations share the same feature, the latter is only appaired with the most likely one, and the others are assigned to the following landmark in their list. This process is repeated until each tree differs from others on its linked feature. If the likelihood of the next landmark to be associated is not greater than some threshold or if a list comes to its end, a new feature is initialized.

2.3.2. Features deletion

Sometimes, previously added features become orphaned, that is they do not correspond to any real tree. To prevent these features to stay in the map, their deletion is carried out together with correspondences establishment. This action is subject to a threshold and the presence of concerned landmarks in the lidar visibility scope.

2.4. *Particles handling*

The common particle deprivation issue of particle filters is counteracted by applying conditional resampling. This shrinking of diversity among particles is also limited by adding gaussian noise to the poses of resampled ones, at the same level as motion noise.

3. Experiments

A model of the SOMA robot, along with a realistic forest environment made of a 30 by 30 meters ground and 16 pine trees have been created, with the aim of conducting simulation experiments (Fig. 3). The software used are Rviz and Gazebo, integrated with ROS.

Fig. 3. Robot model in Gazebo environment

The motor noise experienced in reality is virtually emulated by a ROS node inserted between the steering controller and Gazebo. It consists of a linear gaussian noise added to linear and angular velocity commands respectively.

During all experiments, the robot is manually operated to go straight from one side of the forest to the opposite one. The comparison of the accuracy and update rate of each simulation enables to evaluate how these measures are impacted by the number of particles and the number of trees.

4. Discussion

The accuracy of pose and map estimates has been evaluated with a simulation using 100 particles and realistic motion and observation noises. The Figure 4 shows the current pose and map estimation after 42 seconds of simulation. The same elements are represented as in Figure 1 and the same color code is used. Additionally, the inner space between the two dotted red circles defines the lidar visibility scope.

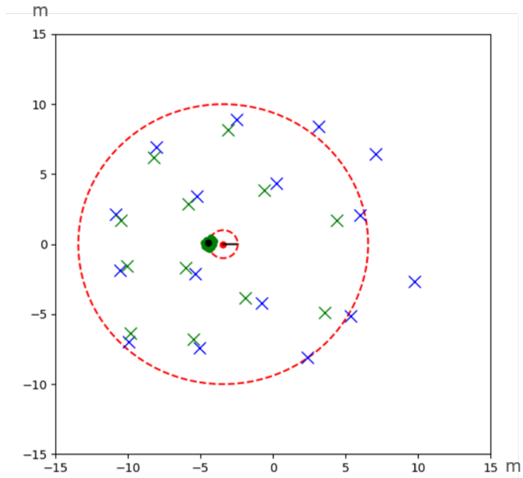

Fig. 4. Display of pose and map estimates at time t = 42 s

Some metrics have been used to measure the efficiency of the implementation. In the experimental conditions described above, the distance between the final real and estimated poses is 2.39 m, and the average distance between each tree and its best corresponding feature in the map is about 0.81 m (Table 1). Five trees over 16 are overrepresented since several landmarks lie around their locations, and one feature in the map is far away from closest trees, leaving it orphaned. Furthermore, an update rate of 1 Hz has been measured during this experiment.

Final position error (m)	2.39
Final map average error (m)	0.81
Redundant features	6
Orphan features	1
Update rate (Hz)	1

Table 1. Simulation quantitative results

As it can be seen on Figure 5, even though correct tracking of the robot is performed on the whole, an increasing distance between the estimated and real poses can be noticed. More precisely, considering the direction of movement, the estimate stays behind the real pose, accumulating late. The delay before taking new odometry data into account can explain some of this gap, but the main reason is most probably to be found in the poor estimation of the coordinates of the trees.

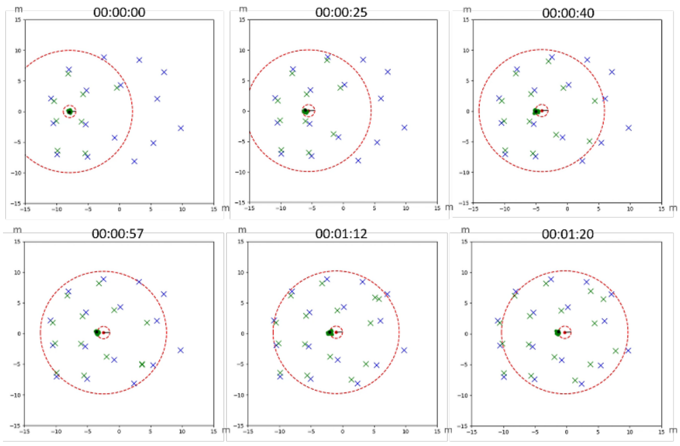

Fig. 5. Display of pose and map estimates at different times

Indeed, these locations are assumed to coincide with the centers of the clusters extracted from the lidar pointcloud. However, since each tree is only seen partly at one time, the former statement does not hold. Besides, the centers of the clusters are constantly changing based on the pose of the robot, breaking the fundamental static world assumption (Fig. 6). Since motion noise is much more significant than observation noise, this is the pose estimate and not the map one which bears the consequences of this issue. Indeed, the estimated pose is updated with the compensation of the virtual movement of trees, moving it in the opposite direction. Because the robot and the extracted centers of trees move in the same direction, the pose estimate tends to be late. To prevent this issue in the future, a circle pattern recognition should be applied to each cluster to get the real centers of the trees.

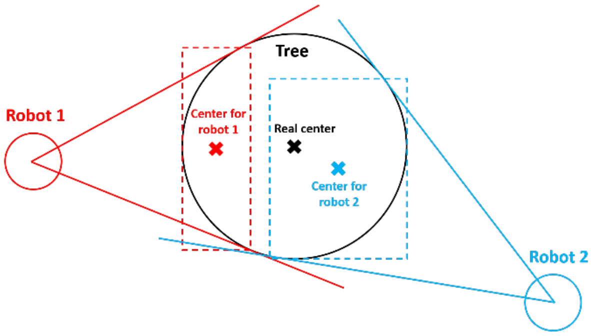

Fig. 6. Centers of clusters depending on the pose of the robot

4.1. *Effect of the number of particles*

The overall accuracy of the proposed approach does not show a substantial dependence with the number of particles used. Actually, only five particles seem to be enough to output a precise pose and map at the end of the experiment. Nevertheless, the update rate suffers from a too high number of particles (Table 2). Finally, the extension of the particles set does not alleviate the systematic shift detailed in the last section.

	Accuracy	Update rate
Effect of increasing the number of particles	None	Decrease

Table 2. Effect of the number of particles on accuracy and update rate

4.2. *Effect of the number of trees*

The late of pose estimate is more considerable when the robot is close to trees, as a smaller part of them is captured by the lidar and therefore the centers of the clusters are far from the centers of the trees. Thus, a high density of the forest lowers the accuracy. In addition, as a well-known issue when solving SLAM, a large number of landmarks tends to make data association harder, involving more frequent errors in this process. Regarding update rate, while the total number of trees does not have any impact on it, a denser forest induces a lower update rate, because only the features in the visibility scope are handled (Table 3).

	Accuracy	Update rate
Effect of increasing the size of the forest	Decrease	None
Effect of increasing the density of the forest	Decrease	Decrease

Table 3. Effect of the number of trees on accuracy and update rate

4.3. *Future research*

A countermeasure should be taken in order to prevent the previously described issue regarding the late of pose estimation. With this in mind, performing a circle pattern recognition on each cluster can be considered as an effective remedy.

Even with randomization during the resampling step of FastSLAM, particles deprivation still tarnishes the results and can lead to failures. Since motion noise is surely dominating observation noise in this situation, using the mixture Monte-Carlo Localization (MCL) method,[6] suitable for reducing deprivation in these cases, could be contemplated. This technique derives from the idea of dual MCL where the roles of motion and observation are inversed. The latter is used to set new particles and the former is used to update the weights of the particles. Dual MCL alone yields poor results, but combining classical and dual approaches with some ratio in mixture MCL significantly improves stability.

Finally, only simulations having been realized so far, real experiments should be conducted as well in order to compare the results and entirely assess the presented approach.

References

1. O. Ikuo, “Declining situation of Japanese forestry today and its challenges toward the 21st Century”, Kyoto University Bioresource Economic Research, 1999
2. M. Matsumoto, H. Oka, Y. Mitsuda, S. Hashimoto, C. Kayo, Y. Tsunetsugu, M. Tonosaki, “Potential contributions of forestry and wood use to climate change mitigation in Japan”, Journal of Forest Research, Vol. 21, 2016
3. A. Mowshowitz, A. Tominaga, E. Hayashi, “Robot Navigation in Forest Management,” Journal of Robotics and Mechatronics, Vol.30, No.2, pp. 223-230, 2018
4. M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, “FastSLAM: A Factored Solution to the SLAM Problem”, 2002
5. D. Gale, L. S. Shapley, “College Admissions and the Stability of Marriage”, The American Mathematical Monthly, Vol. 69, No. 1 (Jan., 1962), pp. 9-15
6. S. Thrun, D. Fox, W. Burgard, “Monte carlo localization with mixture proposal distribution”, Proceedings of the AAAI National Conference on Artificial Intelligence, 2000

Authors Introduction

Mr. Sylvain Geiser

He studied Engineering and Computer Science in Ecole des Mines de Nancy, France, between September 2018 and June 2020. He is currently a Master student at Kyushu Institute of Technology, Japan, and conducts research at Hayashi Laboratory.

Dr. Sakmongkon Chumkamon

Dr. Sakmongkon Chumkamon received Doctor of Engineering degree from Kyushu Institute of Technology in 2017. He was a postdoctoral researcher at Guangdong University of Technology in 2017-2019. Presently he is a postdoctoral researcher in Kyushu Institute of Technology since 2019. His research interests include factory automation robots and social robots.

Prof. Ayumu Tominaga

Prof. Ayumu Tominaga is a professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received the Ph.D. (Dr. Eng.) degree from Kyushu Institute of Technology in 2021. His research interests include Intelligent mechanics, Mechanical systems and Perceptual information processing.

Mr. Takumi Tomokawa

He received bachelor degree in Engineering in 2021 from mechanical system engineering, Kyushu Institute of Technology in Japan. He is aquiring the Master degree in Kyushu Institute of Technology.

Prof. Eiji Hayashi

Prof. Eiji Hayashi is a professor in the Department of Intelligent and Control Systems at Kyushu Institute of Technology. He received the Ph.D. (Dr. Eng.) degree from Waseda University in 1996. His research interests include Intelligent mechanics, Mechanical systems and Perceptual information processing. He is a member of The Institute of Electrical and Electronics Engineers (IEEE) and The Japan Society of Mechanical Engineers (JSME).