Adaptive particle filter for localization problem in service robotics

Alexander Heilig[1], *Ilshat* Mamaev[1 *], *Björn* Hein[1] and *Dmitrii* Malov[2]

[1]Karlsruhe Institute of Technology, Institute for Anthropomatics and Robotics - Intelligent Process Control and Robotics Lab (IAR-IPR), Engler-Bunte-Ring 8, 76131 Karlsruhe, Germany
[2]SPIIRAS, Laboratory of Autonomous Robotic Systems, 199178, Saint-Petersburg, 14-th line of V.I., 39, Russia

Abstract. In this paper we present a statistical approach to the likelihood computation and adaptive resampling algorithm for particle filters using low cost ultrasonic sensors in the context of service robotics. This increases the efficiency of the particle filter in the Monte Carlo Localization problem by means of preventing sample impoverishment and ensuring it converges towards the most likely particle and simultaneously keeping less likely ones by systematic resampling. Proposed algorithms were developed in the ROS framework, simulation was done in Gazebo environment. Experiments using a differential drive mobile platform with 4 ultrasonic sensors in the office environment show that our approach provides strong improvement over particle filters with fixed sample sizes.

1 Introduction

Service robotics market is estimated to almost triple by 2022 from the 2016 level [1]. In comparison to industrial robotics, service robots are inexpensive, built from low-cost hardware and need to fulfil softer requirements for accuracy, repeatability and reliability. This also applies to the sensors. For Simultaneous Localization and Mapping applications (SLAM) in the industry traditionally optical systems, such as 2D/3D cameras and LIDAR sensors, are used. Both types of the sensors are not beneficial in the light of the above requirements, additionally cameras potentially threatening customer privacy. On the other hand, ultrasonic range sensors or sonars are quite appealing for service robotics, the price and power consumption are better than those of optical sensors. Moreover, sonars have no problems with transparent materials like glass and do not acquire any personal data. In this paper we focus on the localization problem of a mobile platform with sonar sensors.

To be able to accomplish service tasks, robots must be able to estimate its pose with respect to a fixed reference frame. This process is known as localization problem. Moreover, localization task can be divided into two subclasses: weak and strong localization. The first one deals only with a qualitative pose estimation such as high level spatial reasoning – for example, determining the room in which the robot is located. On the other hand, strong localization provides numerical estimation of the robot pose. We present an approach to strong localization on the global context using sonars as exteroceptive sensors and odometry as proprioceptive ones. The localization problem is solved using particle filter with a statistical approach to the computation of the likelihood and adaptive resampling algorithm. Modelling and simulation implemented in the ROS framework and Gazebo simulation environment. The evaluation of the proposed approach will be also conducted on an experimental setup using differential drive mobile robot in the institute/office environment.

2 Particle Filter Concept

Uncertainty is a linchpin of the most problems robotics faces in the real world applications. Probabilistic robotics is a relatively new approach, which represents uncertainty using the calculus of probability theory [2]. Bayes filter is the most general approach to compute beliefs based on observations, action data and prior probability. However, in terms of the computational tractability, the general Bayes filter is not trackable for continuous state spaces [3]. The classical trackable solution in probabilistic methods is the Kalman Filter [4], this recursive approach to the discrete-data linear filtering problem based on Gaussian filters has been the subject of extensive research in the SLAM. However, due to the use of the normal distribution assumption, Kalman filter comes to its limits dealing with situations with high ambiguity [5]. The latter is especially relevant for ultrasonic sensors due to low resolution, multiple reflexions and low sampling rate.

In 1999, particle filters were introduced in the localization context under the name of Monte Carlo Localization [6]. In this context, particle filters offer a probabilistic approach on how to estimate the state of a robot in a known map. They are based on the assumption that the measurement of external values is

[*] Corresponding author: ilshat.mamaev@kit.edu

related to the internal state of the robot, i.e. its pose. From there on, the particle filter creates semi-random states in the world – particles – which will give an estimation on their respective states likelihood to represent the actual state. This approach is needed, because the robot's odometry diverges over time and loses its accuracy. The estimation over external sensors corrects this error. A particle filter mainly consists of three phases: motion applying, likelihood estimation, particle resampling (Figure 1).

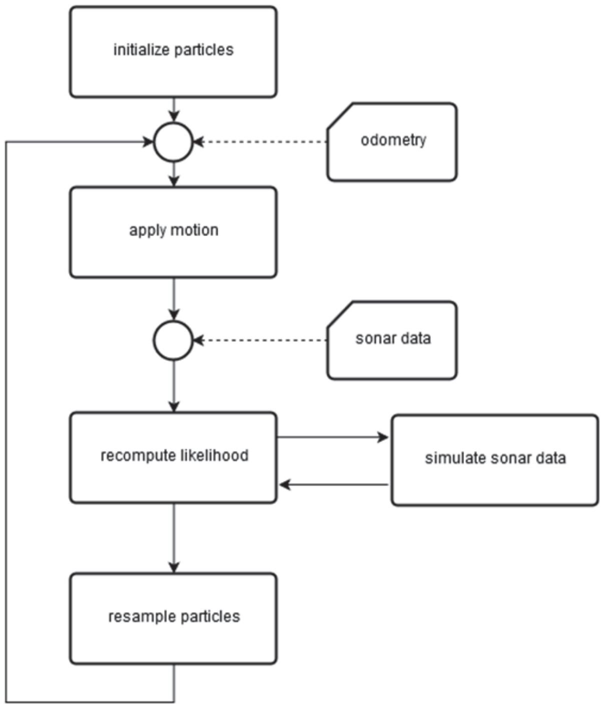

Fig. 1. Particle filter concept and its phases.

This work aims to improve the quality of the particle filter by modifying the computation of likelihood and the resampling algorithm. Particle initialization is uniformly sampled around the map, so the initial state of the robot is unknown. Motion from odometry data is applied in a frequency of 100 Hz to each particle.

2.1 Ultrasonic sensors

In comparison with laser scanners and 3D camera systems, ultrasonic range finders exhibit several shortcomings such as low resolution, cross-talk, multiple reflexions and low sampling rate. Nevertheless, sonars are quite appealing in the context of the service robotics by virtue of the low price, small size and low power consumption. Different research groups have validated the usage of the ultrasonic sensors in localization task [7-9] and even in SLAM [10, 11]. In this paper we exploit the widely used ultrasonic sensor in the robotics community – HC-SR04. The example of the range data from the sensor captured in the test environment is shown in Figure 2.

For the sonar sensor simulation, the hector sonar plugin has been used in Gazebo [12]. This plugin uses the Gazebo integrated ray sensor plugin which scans a cone, originating from the sensor. The plugin then returns the smallest distance it found to simulate sonar data. The simulated sensors were configured to work similar to the real sensors which have been shown to produce reliable results with small variance.

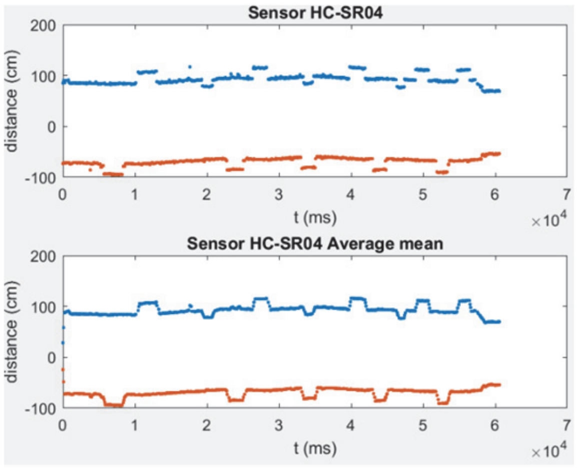

Fig. 2. Raw range measurements (above) and average mean filtered data (below) from HC-SR04 in the test environment.

In the particle filter, sonar sensor measurements must be estimated for each particle. As the number of particles varies from a few hundred to a few thousand, this estimation needs to have low computational cost. The particle filter contains a black and white image in a PNG format representing the real map. Black pixels indicate objects, while white ones indicate free space. Each particle's position is projected onto this bitmap. Using the robot's universal robot description and the sensor attributes, three rays are shot from each particle's sensors. The rays cover the sonar sensor cone's edges as well as the center line. The breadth-first search ray sweep is interrupted, as soon as an object is found by one of the rays, imitating the real sonar's ability to detect only the smallest distance as well as reducing the computational cost. The distance measured for each sensor is then translated from pixels to meters and saved for each particle.

2.2 Likelihood update

After having applied the robot's motion model to all particles, their likelihoods must be updated by including the system observations into the particle filter. The algorithm must assess how well each particle's estimated system observation coincide with the real observation or measurements. The sonar sensors used in this work exhibit a Gaussian noise with standard deviation of 0.005m. Taking this distribution into account, an estimation of likelihood of one estimated sensor measurement to the true measurement can be calculated by

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \quad (1)$$

where μ denotes the actual sensor reading and σ the standard deviation. Taking the arithmetic mean over all probabilities gives a measurement on how likely a particle is to represent the robot state. This calculation has been slightly altered to accommodate differences between particles. Often enough, one sensor reading differs from one particle to another, which contributes only a quarter to the actual likelihood calculated in the end. However, one sensor's reading can be enough to discern a particle with actually high likelihood from a particle with actually low likelihood. For this matter, a particle which has a single sensor reading deviating more than two times the standard deviation σ receives a penalty and has its overall likelihood reduced by 10% for every penalty it got. The resulting likelihood of each sensor of one particle is then saved in a set containing the likelihoods of this particle's ancestors' likelihoods for a fixed size of generations as the most current likelihood. To get a measurement of the particle's final estimation towards the robot state, a weighted average is calculated over the set $\bar{l}$ of current and ancestor likelihoods. A particle's ancestor is defined as the particle it originated from during previous resampling steps. The weighted average is calculated as

$$\boldsymbol{mean(\bar{l}) = \frac{2 * \sum i * l_i}{n * (n+1)}}, \tag{2}$$

$$\boldsymbol{\bar{l} = (l_1, .., l_n), i \in [1, n]}.$$

2.3 Resampling algorithm

Resampling is the last step in particle filter localization algorithms. Particles are deleted, newly generated or altered by taking into consideration previous system observation. This procedure aims to compensate the steadily increasing error of odometry feedback. However, traditional resampling runs into several issues whose consequences can be detrimental for the accuracy of the particle filter.

One of these issues is kidnapping, where the robot is placed somewhere else by an external force which is not measured and from which it must recover to find its actual state again. The traditional particle filter does not recover from such a scenario while the proposed method finds the next possible states, leading to good estimations, but not excluding ambiguities (Fig. 2). Another problem one encounters is the local optimum, i.e. places where the particles get stuck. This usually occurs when estimating the wrong state at a given point in time due to the sensor readings giving high probabilities for this particular state. As the robot moves on, the particles should diverge, as they follow the movement and escape the apparent optimum. However, resampling may lead the particles back to the previous state, resulting in erroneous estimations again.

In this work, the resampling part is based on a systematic resampling approach and extended to increase the accuracy of the state estimation [13]. The particles, sorted by ascending likelihood, are used to generate a set of normalized accumulated likelihoods over all particles, where each accumulation is associated with the respective particle. This approach then generates n ordered equidistant numbers where n equals the size of the resampled particles.

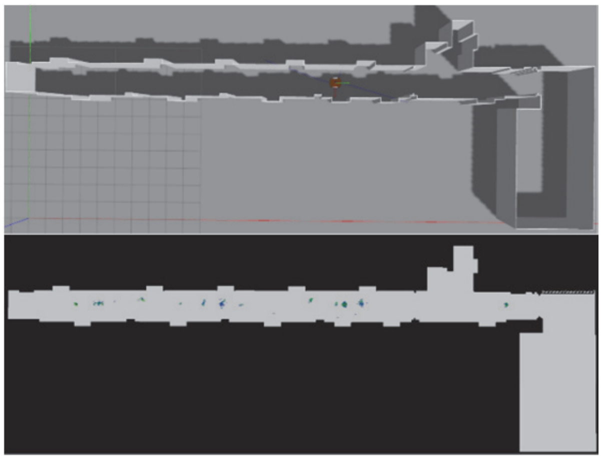

Fig. 3. Ambiguity of states: sensor observations correspond to multiple possible states in the map.

With systematic resampling, one random number $u \in [0, n^{-1}]$ is generated and then used to systematically compute equidistant numbers

$$u_i = \frac{i}{n} + u, i = 2. \tag{3}$$

For each u_i the first particle whose accumulated likelihood satisfies

$$u_i < acc_j, i, j = 1 \tag{4}$$

is drawn for resampling. This accomplishes three things:

1. Prevents sample impoverishment, as the number of resampled particles will be the same as the numbers of previous particles.
2. If there are high likely particles, the algorithm will converge towards them.
3. Systematically ensures that less likely particles will still be picked in contrary to e.g. multinomial resampling.

Drawn particles receive a resampling noise in x, y and θ and are then placed into the new set of particles. The noise added to the drawn particles is restricted to not overstep more than half the distance the motion model had applied to it before. This measurement is taken to ensure the advance of particles along the line of the applied velocity, which in turn guarantees that particle cannot get stuck in local minima. Lastly, the most likely particle is added once more without generating additional noise. This prevents diverging of particles when the odometry error is low. After resampling, the variance of resampling noise and the number of particles is increased, if the best particle's likelihood is underneath a certain threshold. By spreading the particles further out, yet keeping a high density, the particle filter searches for better estimations of the robot state. Starting in the vicinity of the previous particles, small errors in odometry can be detected and corrected. This approach further increases the chance to recover from the kidnapping problem, especially if the robot has not been moved very far from its previous state, i.e. when replacing the robot, because he was in the way of the user. As soon as a state which promises a good estimation is discovered, the number of particles as well as the variance is reduced again. As a result of this approach, only a small number of particles can be used as the default size for the particle filter, thus decreasing the computational cost.

3 Mobile robotic platform "Slamdog"

A low-cost mobile robotic platform "Slamdog" was developed within several student lab projects for teaching and research purposes. This medium sized platform was built upon the commercial children toy vehicle "Feber Dareway 12V" with differential drive kinematics, which was heavily modified: new wheels from BLICKLE for better grip, 14-bit rotatory position sensors AMS AS5047P for odometry, IMU Invense MPU-9150, ultrasonic sensors HC-SR04, IBT H-Bridge etc. Raspberry Pi 3 Model B was used as a low level controller. Furthermore, the platform was expanded with NVIDIA Jetson TX2 development kit as a high level controller, which is mainly used for machine learning tasks and for interfacing with 3D ToF cameras. In this paper, only ultrasonic sensors and the low level controller were used, the localization node was running either on a normal desktop PC or Jetson TX2.

The software for the platform was developed in C++11 under the Robot Operating System (ROS) framework. A ROS hardware driver was implemented as a "Slamdog" ROS node, which provides interfaces for the motor control, ultrasonic sensors, encoders and a buzzer. For simulation of the environment, the robot and the sensors, this work relies on Gazebo 7, for the visualization of particles RViz is used.

4 Evaluation

The high level test scenario for the "Slamdog" was a postman's task: to deliver package inside an office space. In order to be able to do this successfully, it was necessary to navigate in a weakly structured space. The localization problem, presented in this paper, complements the path planning problem within this task. As it was already mentioned one of issue for the localization in this task is the kidnapping of the robot. Figure 4 shows a scenario, where the robot is pushed from its original pose into a different state. This example of small distance kidnapping can be solved by the resampling algorithms dynamically adapting variance and particle numbers.

To further analyze the capabilities of this particle filter approach, different scenarios were simulated multiple times. For each scenario, the distance from the particle filter's best particle as well as the mean distance for all particles and the standard deviation will be reported. Convergence of the particles will be measured by the standard deviation σ and is further defined as $\sigma \leq 0.25$ meters, which roughly equals the radius of the "Slamdog" robot. Measurements were taken every second over twenty seconds. For all scenarios, the initial particle orientation was uniformly distributed over $[-\pi, \pi]$.

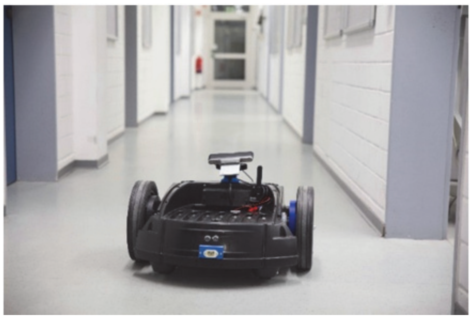
Fig. 4. Slamdog robotic platform in the test environment.

The first scenario "Converged" initialized the particles with a uniformly distribution of 0.5 meters centered at the robot pose to measure the filter's ability to correctly keep a converged state. The second scenario "Kidnapped" is similar to the scenario in Figure 5. The particle filter was initialized with the same distribution as "Converged", but the centered particle pose was offset by 1.0 meter to simulate kidnapping. The third scenario "Ambiguity" was supposed to measure the filter's ability to deal with ambiguities. The robot was placed in the long corridor as depicted in Figure 3 and the particles were initialized over the whole corridor by using a uniformly distribution of 20.0 meters in x- and 8.0 meters in y-direction. The fourth scenario "Diverged" was meant to measure the filter's overall ability to converge to the robot's pose once non-ambiguous measurements exist. The robot was placed at the right end of the long corridor (Fig. 3) and drove to the right. The particles were initially uniformly distributed like in the third scenario. Each experiment was simulated ten times. The results are presented in Table 1.

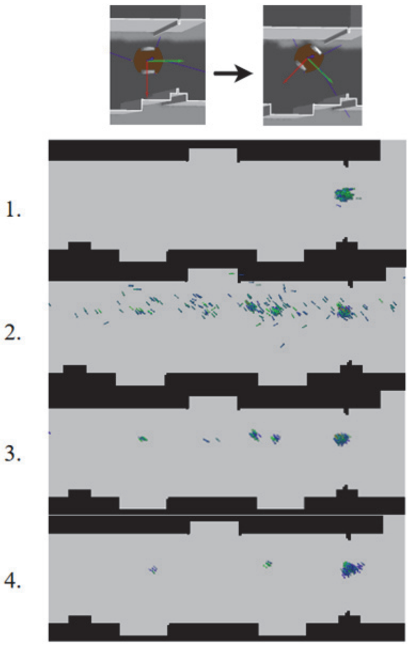

Fig. 5. A kidnapping problem: Robot is forcibly translated and reoriented a small distance from the previous converged state (1). Resampling diverges the state and searches the closer vicinity (2). The particles converge (3) and reach their new best estimation.

Table 1. Results of the particle filter evaluation. Values are measured in distance to the correct robot state in meters for 'best', mean (μ) and standard deviation (σ) or in the time t (in seconds) it took the filter to converge.

Scenarios		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	Mean
Converged	best (m)	0.39	0.21	0.38	0.22	0.42	0.50	0.17	0.30	0.37	0.44	0.34
	μ (m)	0.43	0.41	0.43	0.37	0.48	0.50	0.34	0.41	0.51	0.54	0.44
	σ (m)	0.14	0.14	0.19	0.14	0.12	0.15	0.19	0.13	0.12	0.11	0.14
	t (sec)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Kidnapped	best (m)	0.26	0.08	0.46	0.36	0.15	0.42	0.04	0.34	0.19	0.30	0.26
	μ (m)	0.30	0.34	0.28	0.289	0.21	0.39	0.36	0.38	0.34	0.35	0.32
	σ (m)	0.14	0.12	0.12	0.15	0.11	0.11	0.16	0.12	0.14	0.14	0.13
	t (sec)	3.0	3.0	3.0	4.0	4.0	4.0	2.0	1.0	3.0	3.0	3.0
Ambiguity	best (m)	1.95	1.56	0.91	0.20	1.42	3.45	0.43	1.51	0.20	0.60	1.22
	μ (m)	2.11	1.68	0.59	0.42	2.61	3.23	0.45	1.47	0.37	0.83	1.38
	σ (m)	0.41	0.18	0.88	0.14	1.22	0.16	0.16	0.38	0.14	0.84	0.45
	t (sec)	-	13.0	-	14.0	-	10.0	14.0	-	12.0	-	12.6
Diverged	best (m)	0.38	0.32	0.50	0.24	0.38	0.48	0.50	0.30	0.48	0.22	0.38
	μ (m)	0.38	0.45	0.50	0.43	0.49	0.51	0.63	0.39	0.32	0.35	0.45
	σ (m)	0.14	0.16	0.12	0.14	0.18	0.11	0.13	0.12	0.15	0.17	0.14
	t (sec)	12.0	11.0	11.0	11.0	11.0	7.0	11.0	11.0	12.0	4.0	10.1

The results show that the particle filter is able to keep a fitting state once converged as depicted in "Converged" (see Table 1.). The filter did not expand and converged to a small standard deviation of 0.14 meters overall. The "Kidnapped" scenario illustrated the filter's ability to expand and search its vicinity. The filter needed less than 4 seconds to converge and did so correctly all ten tries. The largest errors were measured in the third scenario. Five times out of ten the particle filter converged in multiple places which shared the same characteristics, i.e. ambiguities (s. Table 1 "Ambiguity" where $\sigma \geq 0.25$). However, three times it has converged correctly. The "Diverged" scenario displays how the particle filter keeps searching the map for a fitting state until at around 11 seconds where salient measurements are taken.

One aspect which can be noted over all the experiments is the range of the best particle found when converged correctly, whose distance to the actual robot spans from 0.08 meters ("Kidnapped", second set) to 0.5 meters (e.g. "Converged", sixth set). This relatively large difference can be explained by the beam angle of the sonar sensors and the distance to the walls on each side in the map. This increases the possible states near the correct state, especially in x-direction and negatively affects finding the best particle. Overall, the particle filter manages to converge correctly in 33 of the 40 experiments undertaken to depict the correct robot state.

5 Conclusions and future work

In this paper we have presented a resampling method, which increases the probability to recover from problems such as stuck particles or kidnapping. This was accomplished by dynamically adapting the configuration of the particle filter to the quality of the state estimation. The algorithm prevents sample impoverishment and converges to high likely states once they are found. The usually high computational cost of estimating the sensor reading for each particle has been simplified while keeping satisfying estimations. Ambiguity of robot states is reduced unless the robot is kidnapped far away to a region that resembles the previous one.

As this particle filter works on comparatively low cost and computational power, the robot system is kept inexpensive. Furthermore, as the main focus lies on applications in the service market, the sonar sensors which have minimal invasiveness regarding privacy and safety, present desired attributes for the sector of service robotics.

In our current implantation we deliberately use only sonar sensors for the localization. This approach can be extended using sensor fusion methods for the sonars, IMU and visual odometry data based on the 2D/3D cameras.

To further analyze the capabilities of this particle filter approach, different scenarios were simulated multiple times. For each scenario, the distance from the particle filter's best particle as well as the mean distance for all particles and the standard deviation will be reported. Convergence of the particles will be measured by the standard deviation σ and is further defined as $\sigma \leq 0.25$ meters, which roughly equals the radius of the "Slamdog" robot. Measurements were taken every second over twenty seconds. For all scenarios, the initial particle orientation was uniformly distributed over $[-\pi, \pi]$.

References

1. *Service Robotics Market and Volume* (Type (Professional Service Robots, Personal and Domestic Service Robots) and Key Players Analysis - Global Forecast to 2022," iGATE RESEARCH, 2017)
2. S. Thrun, W. Burgard, D. Fox, *Probabilistic Robotics* (Cambridge, MA: MIT Press, 2005)
3. A. Burguera, Y. Gonzales, G. Oliver, Advances in Sonar Technology*,* 213–2 32 (2009)
4. R.E. Kalman, Transactions of the ASME--Journal of Basic Engineering, **82**, 35–45(1960)
5. J. Neira, J. Tardós, IEEE Transactions on Robotics and Automation, **17** (6), 890–897(2001)

6. F. Dellaert, D. Fox, W. Burgard, S. Thrun, IEEE International Conference on Robotics and Automation, **2**, 1322–1328 (1999)
7. A. Burguera, Y. Gonzalez, G. Oliver, Sensors, **9**, 10217–10243 (2009)
8. D. Fox, Robotics Research, **22** (12), 985–1003 (2003)
9. A. Großmann, R. Poli, Robotics and Autonomous Systems, **37**, 1–18 (2001)
10. L. D'Alfonso, A. Grano, P. Muraca, P. Pugliese, *16th* International Conference on Advanced Robotics (ICAR), 1–6 (2013)
11. A. Pandey, K. Krishna, H. Hexmoor, International Conference on Integration of Knowledge Intensive Multi-Agent Systems, 283–288 (2007)
12. Hector_gazebo_plugins, Available: http://wiki.ros.org/hector_gazebo_plugins.
13. R. Douc, O. Cappé, Image and Signal Processing and Analysis (ISPA), 64–69 (2005)