We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,900
Open access books available

186,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

An Evolutionary MAP Filter for Mobile Robot Global Localization

L. Moreno[1], S. Garrido[1], M. L. Muñoz[2], and D. Blanco[1]
[1] Robotic's Laboratory, Universidad Carlos III, Madrid
[2] Facultad de Informática, Universidad Politécnica, Madrid
Spain

1. Introduction

This chapter presents a new evolutionary algorithm for Mobile Robot Global Localization. The Evolutive Localization Filter (ELF) presented in this paper is a non linear filter algorithm which is able to solve the global localization problem in a robust and efficient way. The proposed algorithm searches along the configurations space for the best robot pose estimate. The elements of each generation are the set of pose solutions and represent the areas with more probability according to the perception and motion information up to date. The population evolves according to the observation and the motion error derived from the comparison between observed and predicted data obtained from the probabilistic perception and motion model. The algorithm has been tested using a mobile robot with a laser range finder to demonstrate the effectiveness, robustness and computational efficiency of the proposed approach.

Mobile robot localization finds out a robot's coordinates relatives to its environment, assuming that one is provided with a map of the environment. Localization is a key component in navigation and required to execute successfully a trajectory. We can distinguish two different cases: the re-localization case and the global localization case. Re-localization or tracking problem tries to keep track of mobile robot´s pose, where the robot knows its initial position (at least approximately) and therefore has to maintain localized the robot along the given mission. The global localization problem does not assume any knowledge about the robot's initial position and therefore has to globally localize itself.

The two most important aspects that have to be dealt with when designing a localization system is how to represent uncertain information of the environment and the robot's pose. Among the many ways to represent the knowledge about an environment, this article deals with geometrical localization methods and assumes that the environment is modelled geometrically as an occupancy grid map.

In the robot's pose uncertainty representation and estimation techniques, the vast majority of existing algorithms address only the position tracking problem. In this case the small incremental errors produced along the robot motion and the initial knowledge of the robot's pose makes classical approaches such as Kalman filters or Scan Matching techniques applicable. If we consider the robot's pose estimation as a Bayesian recursive problem, Kalman filters estimate posterior distribution of robot's poses conditioned on sensor data.

Based on the Gaussian noise assumption and the Gaussian-distributed initial uncertainty this method represents posteriors distributions by Gaussians. Kalman filter constitutes an efficient solution for re-localization problems. However, the assumption's nature of the uncertainty representation makes Kalman filters not robust in global localization problems. Scan Matching techniques are also an iterative local minimization technique and can not be used for global localization.

Different families of algorithms can solve the global localization problem, some frequently used are: multi-hypothesis Kalman filters, grid-based probabilistic filters and Monte Carlo localization methods. Those methods can be included in a wider scope group of Bayesian estimation methods. Multi-hypothesis Kalman filters (Arras et al., 2002, Austin & Jensfelt, 2000, Jensfelt & Kristensen, 1999, Cox & Leonard, 1994, Roumeliotis et. al, 2000) represent distributions using mixtures of Gaussians, enabling them to keep track of multiple hypothesis, each of which is represented by a separate Gaussian. This solution presents some initialization problems: one of them is the determination of the initial hypotheses (the number can be very high and it is not bounded), this leads the algorithm to a high computational cost at initial stages. Besides, the Kalman filter is essentially a gradient based method and consequently poorly robust if the initial hypothesis is bad or noise assumptions fails. Grid-based localization algorithms (Fox et al., 1999, Burgard et al., 1996, Reuter, 2000) represent distributions by a discrete set of point probabilities distributed over the space of all possible poses. This group of algorithms are capable of representing multi-modal probabilities distributions. A third group is the Monte Carlo localization algorithms (Jensfelt et al., 2000, Thrun et al., 2001, Dellaert et al., 1999). These algorithms represent the probability distribution by means of a set of samples drawn according to the posterior distribution over robot's poses. These algorithms can manage arbitrary noise distributions and non-linearities in the system and observation models. These methods present a high computational cost due to its probabilistic nature requires a high number of samples to draw properly the posterior probability density function. The main advantage is its statistical robustness.

This article presents a localization algorithm based on a non-linear filter called Evolutionary Localization Filter (ELF). ELF solves the global localization robot problem in a robust and efficient way. The algorithm can deal with arbitrary noise distributions and non-linear space state systems. The key idea of ELF is to represent the uncertainty about the robot's pose by a set of possible pose estimates weighted by a fitness function. The state is recursively estimated using set of solutions selected according on the weight associated to each possible solution included in the set. The set of solutions evolve in time to integrate the sensor information and the robot motion information. The adaptation engine of the ELF method is based on an evolutive adaptation mechanism which combines a stochastical gradient search together with probabilistic search to find the most promising pose's candidates.

2. Differential evolutionary filter

The evolutive optimization techniques constitute a series of probabilistic search methods that avoid derivatives or probability density estimations to estimate the best solution to a localization problem. In the method proposed here each individual in the evolutive algorithm will represent a possible solution to the localization problem and the value of the loss function represent the error to explain the perceptual and motion data. The search of this solution is done stochastically employing an evolutive search technique based on the

differential evolution method proposed by Storn and Price (Storn & Price, 2001) for global optimization problems over continuous spaces. The Evolutive Filter uses a a parallel direct search method which utilizes n dimensional parameter vectors $x_i^k = (x_{i,1}^k, ..., x_{i,n}^k)^T$ to point each candidate solution i to the optimization problem at iteration step k. This method utilizes N parameter vectors $\{x_i^k; i = 0, 1, ..., N\}$ as a population for each generation t of the optimization process. Each element of the population set represents a possible solution, but it hasn't associated a probability value to each one (in the particle filter case, each element of the particle set has associated a probability value).

The initial population is chosen randomly to cover the entire parameter space uniformly. In absence of a priori information the entire parameter space has the same probability of containing the optimum parameter vector, and a uniform probability distribution is assumed. The differential evolution filter generates new parameter vectors by adding the weighted difference vector between two population members to a third member. If the resulting vector yields a lower objective function value than a predetermined population member, the newly generated vector replaces the vector with which it was compared; otherwise, the old vector is retained. This basic idea is extended by perturbing an existing vector through the addition of one or more weighted difference vectors to it.

The perturbation scheme generate a variation v according to the following expression,

$$v = x_i^k + L(x_b^k - x_i^k) + F(x_{r_2}^k - x_{r_3}^k) \tag{1}$$

where x_i^k is the parameter vector to be perturbed at iteration k, x_b^k is the best parameter vector of the population at iteration k, $x_{r_2}^k$ and $x_{r_3}^k$ are parameter vectors chosen randomly from the population and are different from running index i. L and F are real and constant factors which controls the amplification of the differential variations $(x_b^k - x_i^k)$ and $(x_{r_2}^k - x_{r_3}^k)$. This expression has two different terms: the $(x_b^k - x_i^k)$ term is a kind of stochastical gradient while $(x_{r_2}^k - x_{r_3}^k)$ is a kind of random search.

In order to increase the diversity of the new generation of parameter vectors, crossover is introduced. Denote by $u_i^k = (u_{i,1}^k, u_{i,2}^k, ..., u_{i,D}^k)^T$ the new parameter vector with

$$u_{i,j}^k = \begin{cases} v_{i,j}^k & \text{if } p_{i,j}^k < \delta \\ x_{i,j}^k & \text{otherwise} \end{cases} \tag{2}$$

where $p_{i,j}^k$ is a randomly chosen value from the interval $[0,1]$ for each parameter j of the population member i at step k and δ is the crossover probability and constitutes the crossover control variable. The random values $p_{i,j}^k$ are made anew for each trial vector i.

To decide whether or not vector u_i^k should become a member of generation $i+1$, the new vector is compared to x_i^k. If vector u_i^k yields a better value for the objective fitness function than x_i^k, then is replaced by u_i^{k+1}; otherwise, the old value x_i^k is retained for the new generation. The general idea of the previous mechanism: mutation, crossover and selection are well known and can be found in literature (Goldberg 1989).
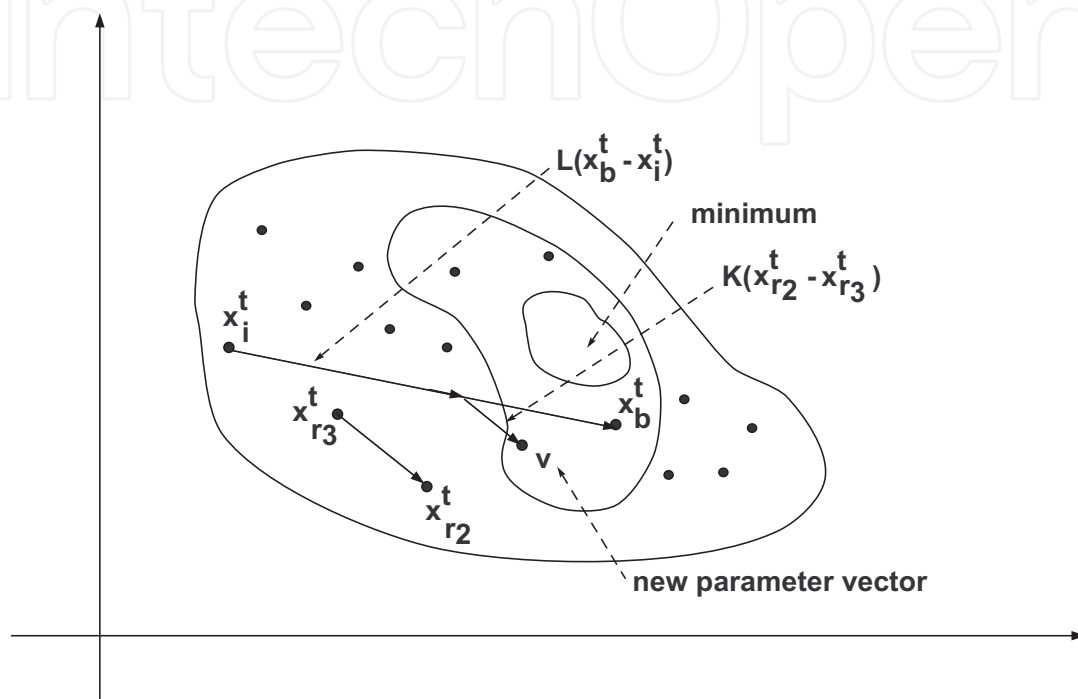

Figure 1. New population member generation

A. Fitness function

Due to we are trying to localize a mobile robot, the natural choice for fitness function is the sum of squared errors function. If the observation vector at time t is $z_t = (z_{1,t}, ..., z_{p,t})^T$ and the predicted observations according the estimated robots pose is $\hat{z}_t = (\hat{z}_{1,t}, ..., \hat{z}_{p,t})^T$ then the penalty function can be stated as

$$L(x_t - \hat{x}_t) = (z_t - \hat{z}_t)^T (z_t - \hat{z}_t) \qquad (3)$$

In the global localization problem there exist some aspects that make this fitness function difficult to manage:

- The range and accuracy of the sensor limits the possibility of discriminate between different poses, leading the fitness function to a high number of global maxima.
- The number of sensors limits the possibility of discriminate between robot's poses leading to multiple global maxima in the fitness function.

- The geometrical similarities in the environment due to the repetition of the space distribution originate the presence of a high number of possible robot's pose solutions to the mean square loss function.

The loss function defined in this way provides us with an optimization mechanism which obtain an unstable parameter solution and by extension an unstable filter. The instability of an evolutive filter based on this penalty function derives from the environment ambiguities. Due to that the evolutive filter can move from one local minimum to other, originating abrupt changes in the pose estimate. This problem comes from the fact we are not using all information we know about the system at the loss function. In fact, we only use the observation model to predict the sensor measurements at each position, and these predicted measurements together with the actual measurements are introduced in the loss function to evaluate the robot's pose estimate. The instability originated by the existence of multiple solutions to the loss function, can not be solved by considering only the available sensor observations. One possibility consists of introducing in the loss function all the information available about the system.

The solution proposed introduced all available information in two ways:

- *State space model information.*

State transition model and observation model are used to estimate the robot's pose and robot's observations next cycle. If we define $x_{i,t}$ as a possible robot's pose solution i obtained in the last iteration of the algorithm at step t. According to the robot motion model the estimated position at step $t+1$ according to the odometric information available will be

$$x_{i,t+1} = g(x_{i,t}, u_t) \tag{4}$$

This information is used for moving the poses of all population members $\{x_i^m\}_t$ obtained in the last iteration m of the evolutive filter at step t. A faster approach, from a computational point of view, consist of estimating only the movement of the best estimate $\hat{x}_{t+1} = g(\hat{x}_t, u_t)$ to estimate a displacement $\hat{d}_t = \hat{x}_{t+1} - \hat{x}_t$ and then to apply to all population member. The initial population at step $t+1$ will be

$$\{x_i^0\}_{t+1} = \{x_i^m\}_t + \hat{d}_t \tag{5}$$

In a similar form the observation model together with the state estimate let us predict the sensor observation values $\hat{z}_t = h(\hat{x}_t)$.

- *Probabilistic information.*

To eliminate the ambiguity, the statistical information available have to be considered. This information is included into the loss function. One possible choice is to include the distance between the estimated robot's pose $\hat{x}_{t+1}$ and each candidate solution x_i^k at iteration k at step ; in the loss function together with the observation error between sensor prediction and sensor measurement.

$$L(\hat{x}_{t+1} - x_i^k) = (\hat{z}_t - z_t)^T Q^{-1}(\hat{z}_t - z_t)$$

$$+[(\hat{x}_{t+1} - x_i^k)^T R^{-1}(\hat{x}_{t+1} - x_i^k)^T] \tag{6}$$

where Q and R are the covariance matrix of the observation and motion noises, and z the sensor data. This loss function includes two effects: the first term considers the observational error, that is the squared distances of sensor observation innovation, and the second term considers the motion innovation. The motion innovation considers the distance between the predicted robot position and the position for a particular solution weighted according a coefficient.

B. The Evolutive Localization Filter algorithm
According with the previous ideas algorithm has the following steps:
- Step 1: Initialization.
The initial set of solutions is calculated and the fitness value associated to each of the points in the state space is evaluated. In the most general case where no information about initial position is available, the initial set of robot's pose solutions is obtained by drawing the robot's poses according to a uniform probability distribution over the state space.
The initial robot's pose estimate is fixed to an initial value.
- Step 2: Evolutive search.
(a) For each element of the set of robot's pose solutions, and according to the map, the expected sensor observations are obtained. The expected observation, the sensor observations, the robot's pose estimate and the robot's pose element are used to evaluate the loss function for each pose element in the set of solutions.
(b) A new generation of perturbed robot's pose solutions are generated according to the perturbation method exposed previously. For each perturbed solution the expected observations are calculated and the loss function evaluated. If the perturbed solution results in a better loss function, this perturbed solution is selected for the following iteration, otherwise the original is maintained.
(c) The crossover operator is applied to the resultant population.
(d) The robot's pose element of the set with lower value of the loss function is marked as best robot's pose estimate. Go to step 2b a given number of iterations.
- Step 3: Updating. The best robot's pose element of the population is used as the updated state estimate and then used in state transition model to predict the new state according to the odometry information.

$$\hat{x}_{t+1} = f(\hat{x}_t, u_t) \tag{7}$$

Then, the displacement is evaluated and the whole population is moved according to this displacement. Then go to step 2.
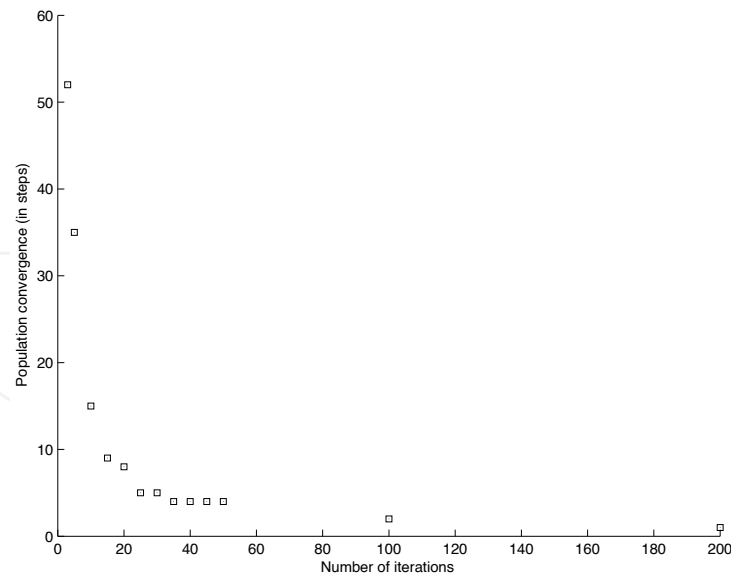
Figure 2. Convergence speed as a function of the iteration number

3. Experimental results

All experiments have been developed in an indoor environment - University laboratories, offices and corridors, and conducted on a B21-RWI mobile vehicle, equipped with laser range finder.

A. Convergence of the algorithm

One fundamental question about evolutionary algorithms is their convergence and their convergence rates: how quickly can the populations come to the individuals (robot's pose solutions) with the highest fitness value?

It has been considered that the algorithm converges to a solution, when all population size is located in a ball of given radius around the best estimate (0.5 meters). Fig. 2 shows the convergence speed of the Evolutive Filter with a set of 500 individuals as a function of the iteration number used for a given trajectory. . As figure 2 suggest, the algorithm proposed is highly efficient in terms of convergence speed. The algorithm behaviour is clearly asymptotical. The speed of convergence grows with the iteration number, but an excessively fast convergence can lead to a local minima. In the tests, the appropriate range of iterations is located between 10 and 30. In this range, the probability of converging to a local minimum is low and the computational effort is reasonable.

Fig. 2 shows a typical convergence process of the Evolutive Localization Filter for a population set of 1000 individuals and 15 iterations of the evolutive filter in each cycle. The robot trajectory starts in the left office of the upper side of the map, goes to the corridor, and continue along the central corridor toward the right side of the map (Fig. 2). It can be noticed that the population individuals tend to concentrate in highly similar areas at the initial pose. After some cycles the algorithm estimate (black circle) reaches the true robot's pose (blue cross) and the population set converges to the true robot's pose. The environment is modelled as an occupancy grid with a cell's size of 12 centimetres. The map scales in figure 2 are expressed in cells.

B. Accuracy and robustness

One striking aspect is the low number of robot's pose solutions required in the population. An appropriate population size for the environment under consideration is 300. The size of

this environment is approximately 865 m^2, and then less than 0.3 elements per square meter is required. In a similar test with Monte Carlo, for this environment are required 10000 samples to localize reliably the mobile robot. Jensfelt tests with a Monte Carlo Localization method (Jensfelt, 2001) for a test environment of 900m² required 10,000 samples. That is 11 samples per square meter.

Fig. 3 shows the performance of the ELF under different sensor noise levels. The ELF algorithm exhibit an relatively constant average error around 6 cm until a sensor noise level of 30%, but even with higher noise levels (50%) is able to localize the robot. This robustness to sensor noise is equivalent to Monte Carlo results shown in (Thrun & Fox, 2001).
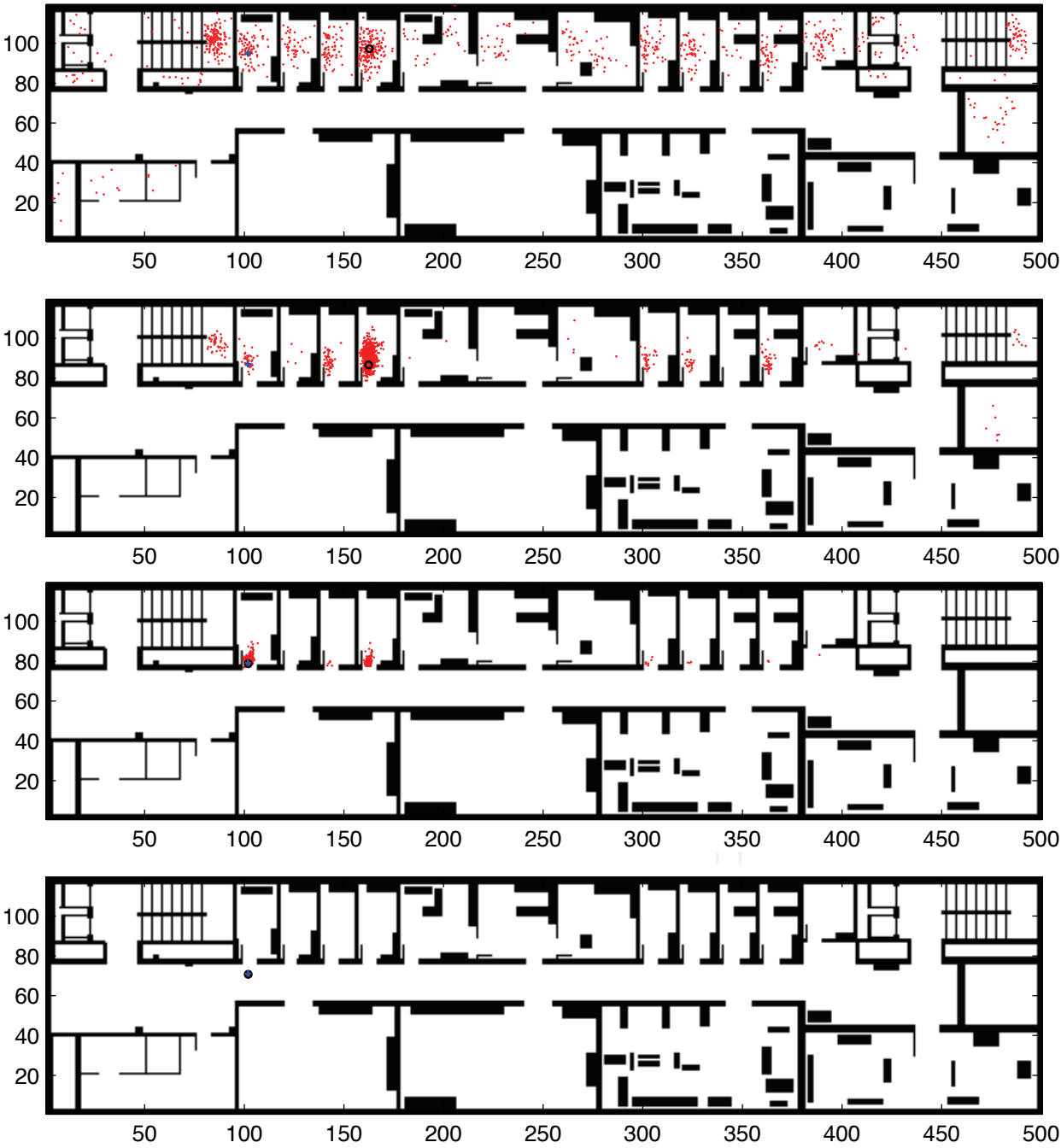
Figure 3. EFL convergence process at cycles: 3, 5, 7 and 9

If we consider the effect of the population size on the accuracy of the algorithm, it can be noticed the minimum effect of the population size on the accuracy. This behaviour is consequence of the search nature of the ELF algorithm. This behaviour differs completely from Monte Carlo results. As noticed by several authors (Doucet, 1998, Liu & Chen, 1998), the basic Monte Carlo filter performs poorly if the proposal distribution, which is used to generate samples, place not enough samples in regions where the desired posterior is large. This problem has practical importance because of time limitations existing in on-line applications.
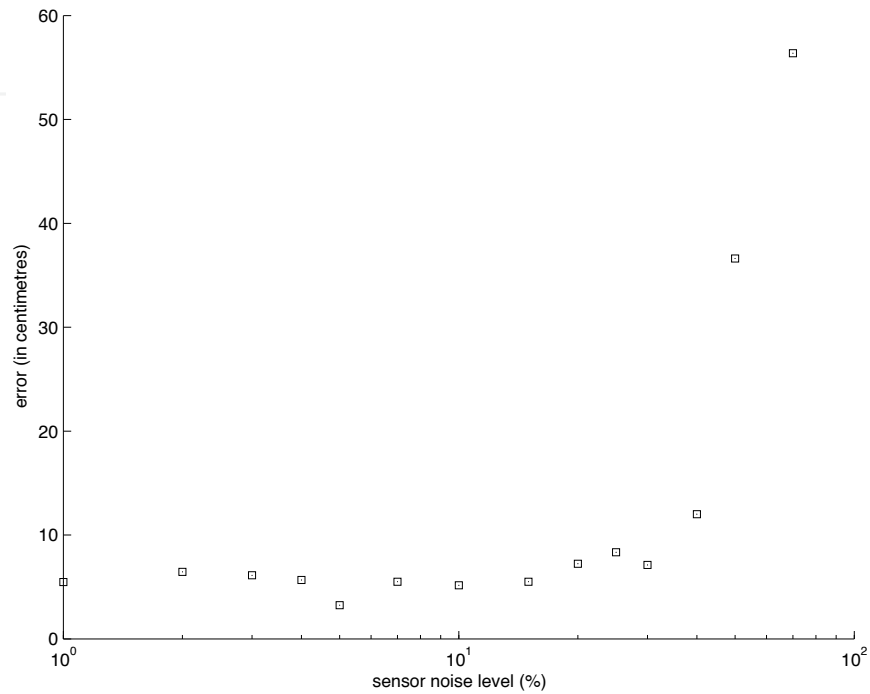
Figure 4. Error of ELF as a function of the sensor noise
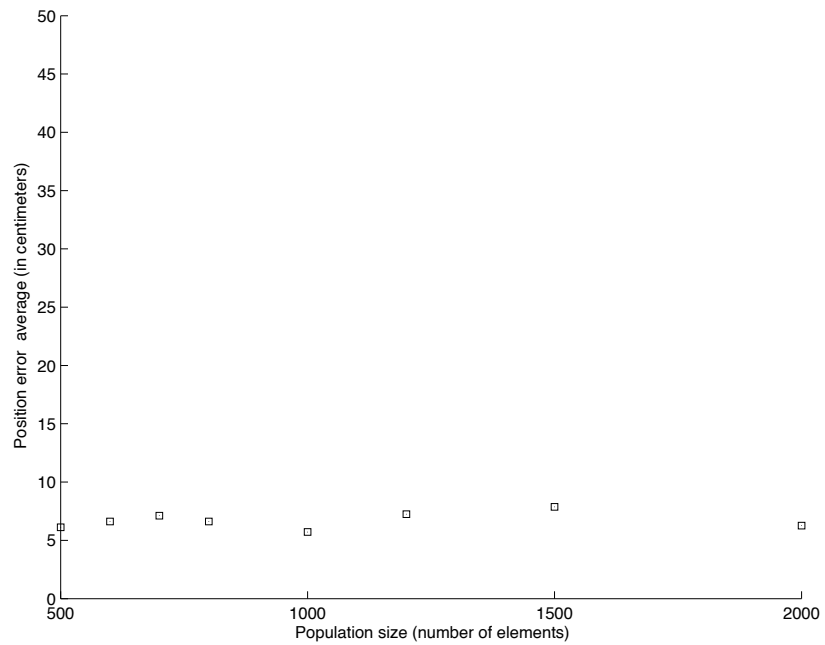
Figure 5. Accuracy as a function of the population size

C. Complexity and computational cost

A final issue addressed in our experiments concerns the running time of ELF algorithm. The absolute time depends on several factors: the computer platform, the observation prediction model and the sensor data, and the population and iterations number. This section illustrates the requirements of ELF. All results reported here were obtained in a PC with an Athlon XP 1800+ processor.

Table 1 shows the time required for ELF to update the estimate value using 60 range laser data, 15 iterations and different population sizes. Table 2 shows the time required for ELF algorithm to update the robot's pose estimate using a fixed population size and changing the iteration´s number. In both cases the behaviour of the algorithm is completely linear. In our experimental localization tests, in a test area of 865m² the best results have been obtained with an initial population of 300 elements and 15 iterations. Once the population converge into a ball of 1 meter radius, the population required to keep the robot localized decrease considerably. In our experiments 50 elements are sufficient. This results shows that even in the initial steps of the ELF algorithm the computational cost is moderate.

Population/Iterations	Time(ms)
1000/15	1703
500/15	859
400/15	704
300/15	520
200/15	359
100/15	180
75/15	141
50/15	94

Table 1. Computation time for a fixed number of iterations

Population/Iter.	Time(ms)
500/5	328
500/10	641
500/15	859
500/20	1125
500/25	1407

Table 2. Computation time for a fixed population of 500 elements

A critical point in any global localization algorithm is the variation in the computational requirements with the environment dimensions. EFL algorithm requirements are proportional to the surface of the environment map. In the experiments done in our laboratory test site, to cope a 900m² area the algorithms requires 300 samples and for an area of 1800m² the samples required are 600. After the population convergence, the samples can be reduced to 30 elements.

4. Conclusions

The proposed algorithm has a range of interesting characteristics:

Evolutive filters can deal with non-linear state space dynamics and noise distributions.

Due to the set of solutions does not try to approximate posterior density distributions, it does not requires any assumptions on the shape of the posterior density as parametric approaches do.

Evolutive filter focus computational resources in the most relevant areas, by addressing the set of solutions to the most interesting areas according to the fitness function obtained.

The number of tentative solutions required in the evolving set is much lower than those required in particle filters, and similarly to those filters the evolving set can be reduced when the algorithm has converged to a reduced area around the best estimate.

The size of the minimum solution's set required to guaranty the convergence of the evolutive filter to the true solution is low.

Due to the stochastic nature of the algorithm search of the best robot's pose estimate the algorithm is able to cope a high level of sensor noise with low degradation of the estimation results.

The algorithm is easy to implement, and the computational cost makes it able to operate on line even in relatively big areas.

The proposed method has been tested under real conditions in a B-21 mobile robot.

Acknowledgment

5. References

Arras, K., Castellanos, J.A., and Siegwart, R. (2002). Feature-based multi-hypothesis localization and tracking for mobile robots using geometric constraints. *Proc. of the Int. Conference on Robotics and Automation ICRA-02*, Washington D.C., USA, pp 1371-1377.

Austin, D.J., and Jensfelt, P. (2000). Using multiple Gaussian hypotheses to represent probability distributions for mobile robot localization. *Proc. of the Int. Conference on Robotics and Automation ICRA-00*, San Francisco, CA, USA, pp 1036-1041.

Burgard, W., Fox, D., Henning, D., and Schmidt, T. (1996). Estimating the absolute position of a mobile robot using position probability grids. *Proc. of the National Conference on Artificial Intelligence AAAI-96*, Portland, Oregon, USA, pp 896-901.

Cox, I.J., Leonard, J.J. (1994). *Modelling a dynamic environment using a Bayesian multi hypothesis approach.* Artificial Intelligence, 66, pp 311-44.

Dellaert, F., Fox, D., Burgard, W., Thrun, S. (1999). Monte Carlo Localization for Mobile Robots. *Proceedings of the 1999 International Conference on Robotics and Automation*, pp. 1322-1328 .

Doucet, A. (1998). On sequential simulation-based methods for Bayesian filtering. *Technical Report CUED/FINFENG/TR 31*, Cambridge University, Dept. of Engineering, Cambridge, UK.

Fox, D., Burgard, W., and Thrun, S. (1999). *Markov localization for mobile robots in dynamic environments.* Journal of Artificial Intelligence Research,11, pp 391-427,1999.

Goldberg, D.E. (1989) *Genetic algorithm in Search, Optimization, and Machine Learning*. Addison Wesley Publishing Company.

Jensfelt, P., Kristensen, S. (1999). Active global localisation for a mobile robot using multiple hypothesis tracking. *Proc. IJCAI Workshop on Reasoning with Uncertainty in Robot Navigation*, pp 13-22, Stockholm, Sweden.

Jensfelt, P., Wijk, O., Austin, D.J. and Andersson, M. (2000). Experiments on augmenting condensation for mobile robot localization. *Proc. of the Int. Conference on Robotics and Automation ICRA-00*, San Francisco, CA, USA, pp 2518-2524.

Jensfelt, P. (2001). *Approaches to mobile robot localization in indoor environments*. Doctoral Thesis, Royal Institute of Technology, Sweden.

Liu, J., Chen, R. (1998). Sequential *Monte Carlo methods for dynamic systems.* Journal Amer. Statis. Assoc., 93, pp 1032-1044.

Reuter, J. (2000). Mobile robot self-localization using PDAB. *Proc. IEEE International Conference on Robotics and Automation (ICRA-2000)*, San Francisco, C.A.

Roumeliotis, S.I., Bekey, G.A. (2000). Bayesian estimation and Kalman filtering: A unified framework for mobile robot localization, in *Proc. IEEE International Conference on Robotics and Automation (ICRA-2000)* ,San Francisco,, pp 2985-2992.

Storn, R. and Price, K. (1995). Differential Evolution- A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-01.

Thrun, S., Fox, D., Burgard, W. and Dellaert, F. (2001). *Robust Monte Carlo localization for mobile robots. Artificial Intelligence*, 128, pp 99-141.
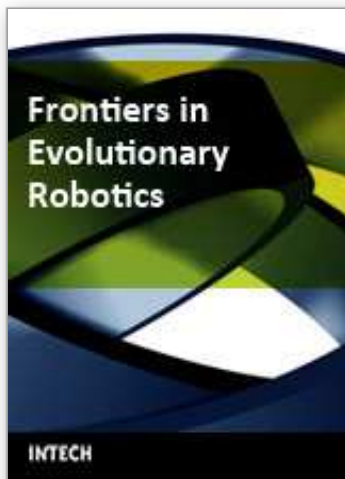
Frontiers in Evolutionary Robotics

Edited by Hitoshi Iba

ISBN 978-3-902613-19-6

Hard cover, 596 pages

Publisher I-Tech Education and Publishing

Published online 01, April, 2008

Published in print edition April, 2008

This book presented techniques and experimental results which have been pursued for the purpose of evolutionary robotics. Evolutionary robotics is a new method for the automatic creation of autonomous robots. When executing tasks by autonomous robots, we can make the robot learn what to do so as to complete the task from interactions with its environment, but not manually pre-program for all situations. Many researchers have been studying the techniques for evolutionary robotics by using Evolutionary Computation (EC), such as Genetic Algorithms (GA) or Genetic Programming (GP). Their goal is to clarify the applicability of the evolutionary approach to the real-robot learning, especially, in view of the adaptive robot behavior as well as the robustness to noisy and dynamic environments. For this purpose, authors in this book explain a variety of real robots in different fields. For instance, in a multi-robot system, several robots simultaneously work to achieve a common goal via interaction; their behaviors can only emerge as a result of evolution and interaction. How to learn such behaviors is a central issue of Distributed Artificial Intelligence (DAI), which has recently attracted much attention. This book addresses the issue in the context of a multi-robot system, in which multiple robots are evolved using EC to solve a cooperative task. Since directly using EC to generate a program of complex behaviors is often very difficult, a number of extensions to basic EC are proposed in this book so as to solve these control problems of the robot.

INTECH
open science | open minds

Fax: +385 (51) 686 166   Fax: +86-21-62489821
www.intechopen.com