5

Global Localization based on a Rejection Differential Evolution Filter

M.L. Muñoz[1], L. Moreno[2], D. Blanco[2] and S. Garrido[2]
[1]*Facultad de Informática. Politechnical University of Madrid*
[2] *Robotics Lab. Carlos III University of Madrid*
Spain

1. Introduction

Autonomous systems are able to move from one point to another in a given environment because they can solve two basic problems: the localization problem and the navigation problem. The localization purpose is to determine the current pose of the autonomous robot or system and the navigation purpose is to find out a feasible path from the current pose to the goal point that avoids any obstacle present in the environment. Obviously, without a reliable localization system it is not possible to solve the navigation problem. Both problems are among the oldest problems in human travels and have motivated a considerable amount of technological advances in human history. They are also present in robot motion around the environment and have also motivated a considerable research effort to solve them in an efficient way.

The localization problem can be addressed in two main ways: on one hand, we have positioning systems and, on the other hand, we have self-localization systems. The *positioning systems* use external emitters (beacons) that are detected by on-board systems or an emitter located on board and several external receivers together with a communication system to send the robot the estimated pose. They use different variants of triangulation methods to estimate the robot pose at a given time. Different positioning systems can be found. The best known is the Global Positioning Systems (GPS) based on satellites around Earth and able to provide a localization in outdoor environments. For indoor environments the problem is more complex due to a high number of emitters and/or receivers required to obtain a complete coverage of the working area. Radio (Wifi and Zigbee), vision, and ultrasound-based systems are active research fields and have achieved an interesting development level, but these technologies depend strongly on the emitters and/or receivers distribution in the building. The positioning systems require to know the location of the emitters but they do not require to have an explicit map of the environment. Obviously, an implicit map is required at least to determine the distribution of the emitters or receivers along the environment. The global complexity of these indoor positioning systems is their weakest point, but it can be very interesting when the number of robots working in a given area is high. The *self-localization systems* use sensing systems located on board the vehicle and do not require any external system. Typical examples are ultrasound, laser, or vision-based localization systems where the emitter and the

receiver are located on the robot. This approach requires a map of the environment (predefined or learned) in order to estimate the robot pose. This chapter will focus on self-location systems, which are a little more autonomous than positioning systems, particularly in indoor environments linked to a mapping learning system.

The self-localization systems solve the pose estimation problem from two different initial situations:

• *Re-localization systems* (also called tracking systems): they try to keep tracking the mobile robot's pose, assuming the robot knows its initial position (at least approximately). Therefore, the self-localization system has to maintain the robot localized along the given mission. The majority of existing algorithms address only the re-localization problem because its mathematical treatment is less complex and, from a practical point of view, it is relatively simple to provide to the robot an initial pose. In this case, the small incremental errors produced along the robot motion and the initial knowledge of the robot's pose make classical approaches such as Kalman filters applicable. The Kalman filter for robot re-localization was introduced in the Eighties (Crowley, 1989; Cox, 1991; Leonard & Durrant-White, 1992; Jensfelt & Krinstensen, 1999) and has been extensively used. This type of filter constitutes a very efficient solution to the re-localization problem. However, the assumptions nature of the uncertainty representation makes Kalman filters not robust in global localization problems.

• *Global localization systems*: they do not assume any a priori knowledge about the robot's initial pose and therefore, they have to estimate the robot's pose globally. This problem has proven to be much more difficult to solve because the search space requires to use global techniques to explore or to integrate the received information until the pose converge to a unique solution.

From a mathematical point of view, the global localization problem can be solved using two different approaches: *Bayesian-based estimation methods* and *optimization-based methods*. In the first approach, *Bayesian methods* integrate all existent probabilistic information (sensor and motion information) into the posterior probability density at each motion-perception cycle, and the point estimate is posteriorly obtained as the state with bigger posterior probability density. Thus, these methods concentrate on the accurate modeling of the posterior probability density as a way to represent the most feasible hypothetical areas and their probabilities. At the convergence point the probability distribution is concentrated in a small area. This group of solutions has been extensively studied and the vast majority of current methods can be included here. Monte Carlo localization methods (Jensfelt et al., 2000) are purely Bayesian methods where the posterior probability distribution is modeled explicitly through the density obtained by the spatial distributions of particles (points with a given probability) along the search space. Other methods can be considered quasi-Bayesian, such as multi-hypotheses Kalman filters (Arras et al., 2002; Austin & Jensfelt, 2000; Jensfelt & Krinstensen, 1999; Cox & Leonard, 1994; Roumeliotis & Bekey, 2000), grid-based probabilistic filters (Fox et al., 1999; Burgard et al., 1996; Reuter, 2000) and, other hybrid solutions where the posterior probability distribution is modeled implicitly (Dellaert et al., 1999; Thrun et al., 2001). Multi-hypotheses Kalman filters are not completely Bayesian because, even if they maintain a set of multi-hypotheses, each of them with an associated Gaussian probability whose aggregated probability distribution can model the posterior, they do not operate on a pure Bayesian way since they use a decision tree search mechanism based on geometrical constraints together with probabilistic attributes to manage the global data association problem.

In the second approach, the *optimization-based methods* use also all existent probabilistic information to obtain a loss function that is minimized at each motion-perception cycle, and the point estimate is the point with lowest value of the loss function. Among the optimization-based approaches we can find differential evolution filters (Moreno et al., 2006) and particle swarm optimization filters (Vahdat et al., 2007). These groups of methods have been more recently used and, only in the last years, have been able to give efficient solutions to the problem. This chapter presents a solution to the global localization problem based on a modified version of the Evolutionary Localization Filter able to deal with the problems introduced by observation and motion noise in the optimization process. The method takes advantage of the capability of the Differential Evolution method to find the minima in complex global optimization problems by using a stochastic search approach.

2. Localization problem formulation

The robot's pose $(x,y,\theta)^T$ at time t will be denoted by x_t, and the data up to time t by Y_t. The posterior probability distribution according to this notation can be written as $p(x_t \mid Y_t, M)$, where M is the environment model which is known. To alleviate the notation, the term M is not included in the following expressions, $p(x_t \mid Y_t)$. The sensor data typically comes from two different sources: motion sensors which provide data related to change of the situation (e.g., odometer readings) and perception sensors which provide data related to environment (e.g., camera images, laser range scans, ultrasound measures). We refer to the former as motions u_i and to the latter as observations z_i. Motion $u(t-1)$ refers to the robot displacement in the time interval $[t-1,t]$ as a consequence of the control command given at time $t-1$. We will consider that both types of data arrives alternatively, $Y_t = \{z_0, u_0, \ldots, z_{t-1}, u_{t-1}, z_t\}$. These sensor data can be divided in two groups of data $Y_t \equiv \{Z_t, U_{t-1}\}$ where $Z_t = \{z_0, \ldots, z_t\}$ contains the perception sensor measurements and $U_{t-1} = \{u_0, \ldots, u_{t-1}\}$ contains the odometric information. To estimate the posterior distribution $p(x_t \mid Y_t)$, probabilistic approaches resort to the *Markov assumption*, which states that future states only depend of the knowledge of the current state and not on how the robot got there, that is, they are independent of past states.

From a Bayesian point of view, the global localization problem seeks to estimate the pose which maximizes the a posteriori probability density. This problem consists of two linked problems. On one hand, the integration of the probabilistic information available into the a posteriori probability density function of each state, given the set of motions, the set of measures and the a priori environment map of the environment. On the other hand an optimization problem to determine the point $\hat{x}_t^{MAP}$ with maximum a posteriori probability density at a given time.

$$\begin{aligned}
\hat{x}_t^{MAP} &= \arg\max_x p(x_t \mid Y_t) \\
&= \arg\max_x p(z_t \mid x_t, u_{t-1}, Y_{t-1}) p(x_t \mid x_{t-1}, u_{t-1}, Y_{t-1}) \\
&= \arg\max_x p(z_t \mid x_t) p(x_t \mid x_{t-1}, u_{t-1}) p(x_{t-1} \mid Y_{t-1}) \\
&= \arg\max_x \prod_{i=1}^{t} p(z_i \mid x_i) \prod_{i=1}^{t} p(x_i \mid x_{i-1}, u_{t-1}) p(x_0)
\end{aligned} \tag{1}$$

This expression requires to specify $p(x_t \mid x_{t-1}, u_{t-1})$ and $p(z_t \mid x_t)$. Where $p(z_t \mid x_t)$ expresses the probability density function for the observation z_t, given the state x_t and an observation noise e, and $p(x_t \mid x_{t-1}, u_{t-1})$ indicates the probability density function for the motion noise v. The expression (1) can be reformulated in an equivalent and more convenient form by taking logarithms:

$$\max_x [\sum_{i=1}^{t} \log p_e(z_i \mid x_i) + \sum_{i=1}^{t} \log p_v(x_i \mid x_{i-1}, u_{i-1}) + \log p(x_0)] \quad (2)$$

In general, the calculation of estimates for this optimization problem have no explicit analytical solutions for nonlinear and non-Gaussian models, and have to be iteratively solved to avoid the difficulties included in the optimization problem. These difficulties derive from the following aspects:

1. It is highly non-linear. Non-linearities due to motion and perception functions are propagated through the a posteriori robot pose probability density function.

2. Environment symmetries make the objective function to maximize multi-modal. At initial stages the objective function admits a high number of solutions, even with the same maximum value. That happens in highly symmetric environments, such as typical offices buildings. The reason can be noticed in (2), where the second term p_v is a constant in absence of robot's motion and the third term $p(x_0)$ is also constant in absence of initial pose information. This leads to an objective function $\max_x \sum_{i=1}^{t} \log p_e(z_i \mid x_i)$ which only depends on observations and has potentially multiple maxima in highly regular environments.

3. Another source of symmetries is originated by sensor limitations. The range and angular resolution of the sensor adds observational symmetries. Besides, some specific robot's poses can originate observational limitations which adds symmetries (e.g., a robot closes to a corner and looking at the corner).

In order to solve (2), a set of candidate estimates have to be initially generated, maintained or pruned according to the new observation and motion information included in the objective function. The problem is simplified in case the initial probability distribution is Gaussian, because the problem becomes uni-modal and then, it is possible to obtain, even analytically, an estimate (due to the problem can be converted into a quadratic minimization problem, if non linear motion and observation models can be approximated by a linear Taylor series expansion about the current estimate $\hat{x}_t$). This situation leads us to the well known Extended Kalman Filter solution of the position tracking problem.

We will use the notation $f_0(x)$ to refer the objective function to maximize. The problem of finding an x that maximizes $f_0(x)$ among all x that satisfy the conditions $x_{t+1} = f(x_t, u_t) + v_t$ and $z_t = h(x_t) + e_t$ is limited to finding the optimal value within the set of all feasible points. A pose is feasible if it satisfies the constraints $f()$ and $h()$. In the problem under consideration, there exist, at least at initial stages, multiple optimal values. Thus, the methods to solve the problem require to be able to manage a set of solutions. The Bayesian methods use the a posteriori probability density function to do that, as was previously commented. The method proposed here uses a different approach. The idea is to maintain a set of feasible solutions to the localization problem, and let this set evolve towards optimal values according to the observed motion and perception data.

2.1 Recursive formulation

The MAP estimate formulated as an optimization problem subject to conditions, in equation (2), is not practical from a computational point of view. To implement a global localization algorithm in a robot, a recursive formulation is required. The objective function $f_0(x_t)$ can be expressed recursively in the following way:

$$\begin{aligned} f_0(x_t) &= \sum_{i=1}^{t} \log p_e(z_i \mid x_i) + \sum_{i=1}^{t} \log p_v(x_i \mid x_{i-1}, u_t) + \log p(x_0) \\ &= \log p_e(z_t \mid x_t) + \sum_{i=1}^{t-1} \log p_e(z_i \mid x_i) \\ &+ \log p_v(x_t \mid x_{t-1}, u_{t-1}, m) + \sum_{i=1}^{t-1} \log p_v(x_i \mid x_{i-1}, u_{t-1}) + \log p(x_0) \\ &= \log p_e(z_t \mid x_t) + \log p_v(x_t \mid x_{t-1}, u_{t-1}) + f_0(x_{t-1}) \end{aligned} \tag{3}$$

If we are able to solve the optimization problem at time $t-1$, and we have a set of sub-optimal solutions which satisfy the optimization problem up to time $t-1$, the MAP optimization problem can be reformulated as

$$\hat{x}_{t-1} = \max_x \log p_v(z_t \mid x_t) + \log p_e(x_t \mid x_{t-1}, u_{t-1}) \tag{4}$$

where $\hat{x}_{t-1}$ is the x which maximize the MAP optimization problem at time $t-1$, and x^*_{t-1} is the population set of sub-optimal solutions at the end of iteration $t-1$. Then solving (4) we will obtain a recursive version of the MAP estimate.

3. Evolutionary Localization Filter algorithm

3.1 Differential Evolution: basic concepts

The algorithm proposed to implement the adaptive evolutive filter is based on the differential evolution method proposed by Storn and Price (Storn & Price, 1995) for global optimization problems over continuous spaces. The Adaptive Evolutionary Localization Filter uses as a basic solution search method, the classical **DE/rand/1/bin** version with some modifications to improve its characteristics in presence of a noisy fitness function.

The **DE/rand/1/bin** uses a parallel direct search method which utilizes n dimensional parameter vectors $x_i^k = (x_{i,1}^k, \ldots, x_{i,n}^k)^T$ to point each candidate solution i to the optimization problem at iteration k for a given time step t. This method utilizes N parameter vectors $\{x_i^k; i = 1, \ldots, N\}$ as a sub-optimal feasible solutions set (population) for each generation t of the optimization process.

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 (see fig. 2).

3.1.1 Differential Perturbation Operation

The perturbation scheme generates a variation v_i^k according to the following expression,

$$v_i^k = x_{r_1}^k + F(x_{r_2}^k - x_{r_3}^k) \tag{5}$$

where $x_{r_1}^k$, $x_{r_2}^k$ and $x_{r_3}^k$ are parameter vectors chosen randomly from the population, different from running index i, and mutually different. F is a real constant factor which controls the amplification of the differential variation $(x_{r_2}^k - x_{r_3}^k)$.

3.1.2 Crossover Operation

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

$$u_{i,j}^k = \begin{cases} v_{i,j}^k & \text{if } p_{i,j}^k < \epsilon \\ x_{i,j}^k & \text{otherwise} \end{cases} \tag{6}$$

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 ε 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.

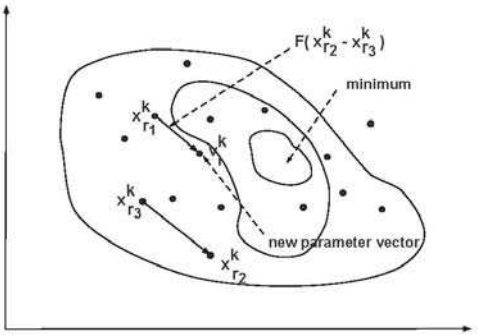


Fig. 1. New population member generation.

3.1.3 Selection Operation

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 value for the objective fitness function better than x_i^k, then is replaced by u_i^k for the new generation; otherwise , the old value x_i^k is retained for the new generation.

3.1.4 Shift Operation

After the DE algorithm has completed its iterations, the points included in the population set x_t^* are moved according to the robot motion model $x_{t+1}^i = f(x_t^i, u_t)$, the candidate pose and the observed odometric data.

3.1.5 Fitness function

According to the optimization problem under consideration, $\max_x(\log p_v(z_t \mid x_t) + \log p_e(x_t \mid x_{t-1}, u_{t-1}))$, the natural choice for fitness function is

$$f_0(x^t) = \log p(z_t \mid x_t) + \log p(x_t \mid x_{t-1}, u_{t-1}) \tag{7}$$

This expression contains two probability densities associated to errors in the motion and observation models (the perception error probability density distribution $p(z_t \mid x_t)$ and the robot's motion error probability density distribution $p(x_t \mid x_{t-1}, u_{t-1})$). A third probability model is used to model the information we have at initial stage about the initial a priori robot's pose $p(x_0)$. This initial probability pose distribution is used at the initial phase to distribute the population set of the ELF algorithm. In case of global localization problem, the initial pose information is null. Then, the population set is distributed according to a uniform probability distribution along the space state.

To compute $p(z_t \mid x_t)$, it is necessary to predict the value to be observed by the sensor, assuming that the robot's pose estimate is known. Let assume the pose estimate is $\hat{x}_t$, the sensor relative angle with respect to the robot axis is α_i and a given environment model m. According to the observation model, the noise-free predicted measurement will be $\hat{z}_{t,i} = h(\hat{x}_t, \alpha_i, m)$ (in our case, $\hat{z}_{t,i}$ is computed using a ray tracing method). Assuming that the measurement error is Gaussian with zero mean and known covariance ($e_{t,i} \approx N(0, \sigma_e)$), the predicted measurement will be the center of the Gaussian probability distribution of the expected distance measured by the α_i sensor when robot is located at x_t. Then the probability of observing $z_{t,i}$ with sensor i can be expressed as

$$p(z_{t,i} \mid \hat{x}_t) = \frac{1}{(2\pi\sigma_e^2)^{1/2}} e^{-1/2\frac{(z_{t,i}-\hat{z}_{t,i})^2}{\sigma_e^2}} \tag{8}$$

The integration of all individual sensor beam probabilities into a joint probability value, assuming conditional independence between the individual measurements, is expressed as

$$p(z_t \mid \hat{x}_t) = \prod_{i=0}^{N_s} p(z_{t,i} \mid \hat{x}_t) = \prod_{i=0}^{N_s} \frac{1}{(2\pi\sigma_e^2)^{1/2}} e^{-1/2\frac{(z_{t,i}-\hat{z}_{t,i})^2}{\sigma_e^2}} \tag{9}$$

where N_s is the number of sensor observations.

The second probability required to calculate the objective function is $p(x_i \mid x_{i-1}, u_{t-1})$. To compute $p(x_i \mid x_{i-1}, u_{t-1})$, we have to predict the robot's pose $\hat{x}_t$ assuming that the robot's

pose estimate is $\hat{x}_{t-1}$ and taking into account the motion command u_{t-1} at that cycle. Let $\hat{x}_t = f(\hat{x}_{t-1}, u_{t-1})$ denote this ideal predicted state. Assuming the motion error is a zero mean with known variance Gaussian probability distribution (that is $v \approx N(0,P)$), this predicted measure will be the center of the Gaussian probability distribution of the expected distance when the robot is located at $\hat{x}_t$. Then, the $p(x_i \mid x_{i-1}, u_{t-1})$ probability can be expressed as

$$p(x_i \mid x_{i-1}, u_{t-1}) = \frac{1}{\sqrt{|P|(2\pi)^n}} e^{-1/2(x_i - \hat{x}_i)P^{-1}(x_i - \hat{x}_i)^T} \tag{10}$$

Introducing the expressions of $p(x_t \mid x_{t-1}, u_{t-1})$ and $p(z_t \mid x_t)$ in (7)

$$\begin{aligned} f_0(x_t) &= \log \prod_{i=0}^{N_s} (2\pi\sigma_e^2)^{-1/2} e^{-\frac{(z_{t,i} - \hat{z}_{t,i})^2}{2\sigma_e^2}} \\ &+ \log(|P|(2\pi)^n)^{-1/2} e^{-\frac{1}{2}(x_i - \hat{x}_i)P^{-1}(x_i - \hat{x}_i)^T} \\ &= \sum_{i=0}^{N_s} \log(2\pi\sigma_e^2)^{-1/2} - \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i})^2}{2\sigma_e^2} \\ &+ \log[(|P|(2\pi)^n)^{-1/2}] - \frac{1}{2}(x_i - \hat{x}_i)P^{-1}(x_i - \hat{x}_i)^T \end{aligned} \tag{11}$$

which can be reduced to find the robot's pose to minimize the following function

$$f'_0(x_t) = \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i})^2}{2\sigma_e^2} + \frac{1}{2}(x_i - \hat{x}_i)P^{-1}(x_i - \hat{x}_i)^T \tag{12}$$

The differential evolutive localization filter will minimize iteratively the fitness function (12) and then, the displacement is evaluated according to the odometric information. The objective fitness function let us notice that the optimization considers the quadratic observation error and the quadratic pose error between the predicted pose and the pose under consideration weighted according its covariance matrix.

4. Difficulties to achieve a robust method

Different problems need to be solved in optimization methods to achieve a reasonable robustness level. Some of this problems are general and related to the global localization problem, and others are related to the nature of the basic algorithm adopted. Among the general problems, we can remark the following ones:

• The lack of a method to determine the significance of a new point apart from the fitness function value. This originates two classes of problems: **premature convergence** to a local minima and, in case of multiple hypotheses situations, the premature elimination of feasible hypotheses. Both situations originate a fail in the convergence to the true global pose. This second situation can be observed in figure 2, where the red points are the pose hypotheses at population set. In this case, the robot is localized at an office exactly equal to others in dimensions and furniture. Due to the fact that the robot's orientation is 270 degrees, it can not

distinguish between offices and the hypothesis should be maintained trough iterations. But, if we let the algorithm iterate, it can be observed how some of the hypotheses are removed. This process can end in one hypothesis. Obviously, this algorithm behavior is not robust.

Fig. 2. Premature hypothesis elimination, initial pose (105, 30, 270), 200 elements in population and σ of 3% of the measured signal.

A traditional solution is to increase the population number to make more difficult the premature elimination of feasible hypotheses and to limit the number of iterations to control the algorithm progress.

• A second important problem is how to determine **the stopping criteria**. Traditional methods are: to fix a predefined iteration number or to stop the algorithm when the fitness function does not improve for a number of iterations. However the use of a predefined iteration number can also lead to premature elimination of hypotheses, as can be observed in the example of figure 3, where, after three motions the algorithm, it has converged prematurely to an incorrect pose. In the figure, the blue cross indicates the best fitness pose of the population set and the magenta points show the observed laser measurements projected over the environment map according to the best pose estimate.
The second idea for the stopping criteria consists on stopping the algorithm if after a predefined number of iterations the best hypothesis is not modified. This stopping criteria is not easy to establish. If the number of iterations without improvement is low, the algorithm can not converge or converge very slowly. This problem can be observed in the example of figure 4 that shows a stopping criteria of 20 cycles without best fitness improvement. In the figure can be notice that the algorithm can not converge properly. This problem can be partially eliminated by using a bigger number of default cycles. In that case, the algorithm does more iterations at each perception-motion cycle but then, we move to the previous case situations where the algorithm converges prematurely to an improper pose. Obviously, we can try to adjust the parameter to each situation, but this adjust depends on the observation and motion noises, on the shape and size of the environment observed at initial pose and, consequently, it is not robust.

Fig. 3. Premature hypothesis elimination, starting pose (305, 30, 270), 200 elements in population and σ of 3% of the measured signal, motion +2.5 cells per cycle in y direction.

• A third problem is the population dispersion. The problem can be perceived in the last image of figure 4, where the population set is dispersed considerably when the robot goes out of the office and passes to the corridor. Since Differential Evolution is a stochastic search method, the pose set spreads along the best fitness areas contained in the stochastic search ball (defined by the possible combinations of three elements stochastically taken from the population set). If the population has not converged, it spreads when the robot moves to an area where many possible poses has a similar fitness value. This problem is originated by the lack of memory of each individual pose of the population set in the algorithm.

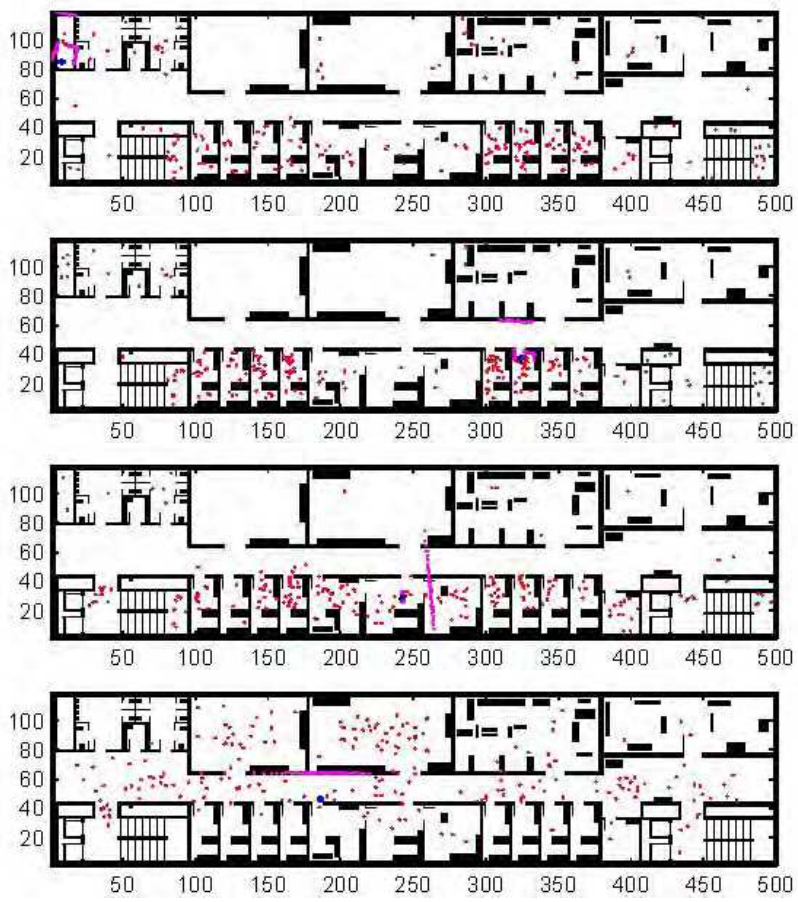



Fig. 4. Convergence failure, starting pose (305, 30, 270), 200 elements in population and σ of 1% of the measured signal, motion +2.5 cells per cycle in y direction.

4.1 Solutions to deal with noisy fitness function problems

The two first problems are originated when a superior candidate solution may be erroneously considered as an inferior solution due to the noise and eliminated from the set

of solutions by the algorithm. Different solutions has been proposed in literature to compensate the noise problem in evolutive algorithms:

• Increasing the population size. It is the most simple way to deal with the noise problem. This technique reduces the probability of eliminating correct hypothesis prematurely, but increases the computational cost of the algorithm. This solution does not solve the dispersion problem.

• Resampling and averaging the fitness function from several samples reduces the error in the fitness. This method assumes as estimate to evaluate the fitness function the sampling mean that has its standard error reduced by $\sqrt{n}$, where n is the number of samples used to estimate the mean. This technique is frequently used by the optimization researchers, but it requires to sample the fitness function repeatedly. This technique can not be used for dynamical systems because these systems do not remain in the same state and consequently, they can not be used for the global localization system problem.

• Thresholding was proposed by Markon (Markon et al., 2001). The idea is to replace an existing candidate solution only when the fitness difference is larger than a given threshold τ. This method requires to calculate the threshold value, which depends on the variance of the noise and the fitness distance to optimal fitness value. This mechanism requires to increase the number of iterations, since the level of candidate solutions rejected increases.

• Estimation of the true fitness. This idea was suggested by Branke (Branke et al., 2001). He proposes to estimate an individual fitness using a local regression of the fitness of the neighboring individuals. The underlying assumptions of this method are: that the true fitness function can be locally approximated by a low polynomial function, that the variance in a local neighborhood is constant, and the noise is normally distributed.

The third problem (dispersion) requires a different approach and has not be widely studied in literature (perhaps because it appears mostly in dynamical system subject to noise).

5. Rejection Differential Evolution Filter

The solution adopted in this work use three main mechanisms to improve the robustness and efficiency of the basic DE algorithm to solve the global localization problem. These mechanisms are:

1. A **threshold rejection band** to avoid the premature elimination of solutions. This mechanism decreases the eagerness of the algorithm, allowing it to eliminate a candidate solution from the set only when the offspring candidate is significatively better from a statistical point of view.

2. An **stopping criteria** based on the expected fitness value to stop the algorithm iterations in a statistically equivalent point. This idea will let the algorithm iterate as much as possible to obtain a fast convergence towards the solution if there is a statistical improvement between iterations or to stop very fast if no statistical improvement is obtained.

3. Adjustment of the **perturbation amplification factor** F. This mechanism tries to maintain a high amplification factor while the population evolves in the first perception cycle to the most the promising areas (a wide scope search is required) and then to limit the algorithm search scope when the population set is distributed in the most feasible areas.

5.1 Threshold determination

The fitness function to minimize is given by expression (12). For a given candidate x_t^j the fitness function value is given by

$$f_0^j(x_t^j) = \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i}^j)^2}{2\sigma_e^2} + \frac{1}{2}(x_t^j - \hat{x}_t)P^{-1}(x_t^j - \hat{x}_t)^T \quad (13)$$

where $z_{t,i}$ is the measure given by the range scan sensor at angle α_i at cycle t, $\hat{z}_{t,i}^j$ is the estimated observation for the candidate robot's pose x_t^j, and x_t is the pose estimate (if it exists at cycle t). The second term of the expression depends on the robot pose estimate $\hat{x}_t$ that is not known at initial step, and it is neglected until a unique pose estimate is obtained (that happens when all population has converged to a limited area around the best pose estimate). The fitness function before the convergence point information takes the following form:

$$f_0^j(x_t^j) = \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i}^j)^2}{2\sigma_e^2} = \frac{1}{2}\sum_{i=0}^{N_s} \frac{v_{t,i}^2}{\sigma_e^2} \quad (14)$$

where $v_{t,i} = (z_{t,i} - \hat{z}_{t,i}^j)$ represents the discrepancy between the observed and the predicted value of the sensor data. To estimate the expected noise band for the fitness function, we need to calculate the expected value for $E[f_0]$ when the pose under evaluation is the true one. The term $\sum_{i=0}^{N_s} v_{t,i}^2/\sigma_e^2$ where $v_{t,i}/\sigma_e^2$ are standard normal random variables $N(0,1)$ is a chi-square distribution with N_s degrees of freedom. This distribution is well known and it has mean N_s and variance $2N_s$. Then, the expected minimum fitness value will be

$$E[f_0] = \int_{-\infty}^{+\infty} f_0(v)p(v)dv = N_s/2 \quad (15)$$

That means that, even if the pose we are considering was the true robot's pose, the expected fitness function value would be $N_s/2$ due to observation errors produced at the perception time. If two candidate poses x_1 and x'_1 are compared at a given iteration time, the question is: when can we consider there exists a reasonably evidence that candidate pose x_1 is better than x'_1 ?. In the tests, different values for the threshold rejection level have been simulated. To maintain the elitism in the method, one exception has been introduced (a pose candidate with a fitness better than the best pose existent up to that moment will always pass to the following iteration). That exception consists of selecting the best pose obtained for the next iteration population, independently of the rejection threshold.

5.2 Stopping condition

A classical problem in optimization methods is how to determine a stopping condition. This problem can be considered in different ways: limiting the number of iterations, iterating until a pre-specified accuracy is obtained or iterating until no additional improvement is obtained. But in case of noisy fitness problems, those conditions are not appropriate.
Assuming that the fitness function is a chi-square with N_s degrees of freedom, it is possible to obtain the p-quantile function value with a pre-specified p value of probability, or in

other words, the fitness function value f_{1-p} that has $1-p$ probability of being inferior to the fitness function value at any perception cycle. Quantile values for some pre-specified probability values and degrees of freedom can be found in statistics literature.

5.3 Amplification factor

The previous mechanisms improve greatly the robustness but they do not exploit the local convergence. This effect is clearly evident in office buildings where many offices have the same dimensions, which originates a multiple convergence areas. Once the robot gets out of the office to a corridor, if factor F is maintained high, the population spreads along the corridor areas.

To avoid this problem, the amplification factor is initialized at $F = 0.99$ in the first iteration of the observation cycle and, after the first perception cycle, F is decreased to a low value, $F = 0.05$. This tends to keep the search area of the algorithm at initial perception cycle as wide as possible and, once the algorithm has localized the most feasible areas, the amplification factor is decreased to a low value to concentrate the exploration in the surroundings of the previous areas avoiding an unnecessary dispersion of the population.

6. Convergence results

To test the algorithm characteristics, a simulated environment has been considered (figure 5). This environment is similar to many office indoor areas. All offices are localized along the central corridor. The offices localized on the upper part of the figure have the same length in y dimension and an x length progressively decreasing (in one cell) from offices localized on the left side of the figure to those located on the right side. On the contrary, offices localized on the lower part of the figure are of exactly the same dimensions and appearance. The offices localized on the upper and lower corners of the environment have similar dimensions but doors are localized on different sides.

6.1 Test 1

The first test tries to determine the capability of the algorithm to localize the robot when it is localized at a distinguishable pose. The true robot's position is $(x, y, \theta)^T = (60, 60, 0)^T$ and the variance considered for each measurement in the simulation is of 3% of the measured signal, which is relatively noisy compared with real laser scanners. The population set used in the test is of 100 elements.

In the test example of figure 5, the stopping condition happens at iteration 334. At that point the estimated pose is $(60.232, 60.098, 359.839)$ (units are in cells and degrees). The size of the cell considered for the map is of 12 cm, which corresponds to an estimation error of 2.79 cm in x dimension, 1.182 cm in y and 0.16 degrees in orientation, which is quite accurate for the noise level and for one perception cycle. In figure 5, the red points indicate the candidate poses position, the magenta points represent the points observed in the environment according to the best pose obtained, and the blue cross represents the best pose obtained. If we increase the noise level, the number of feasible points at the end of the first perception cycle tends to increase, since to the noise level tends to make the disambiguation capability of the method more difficult.

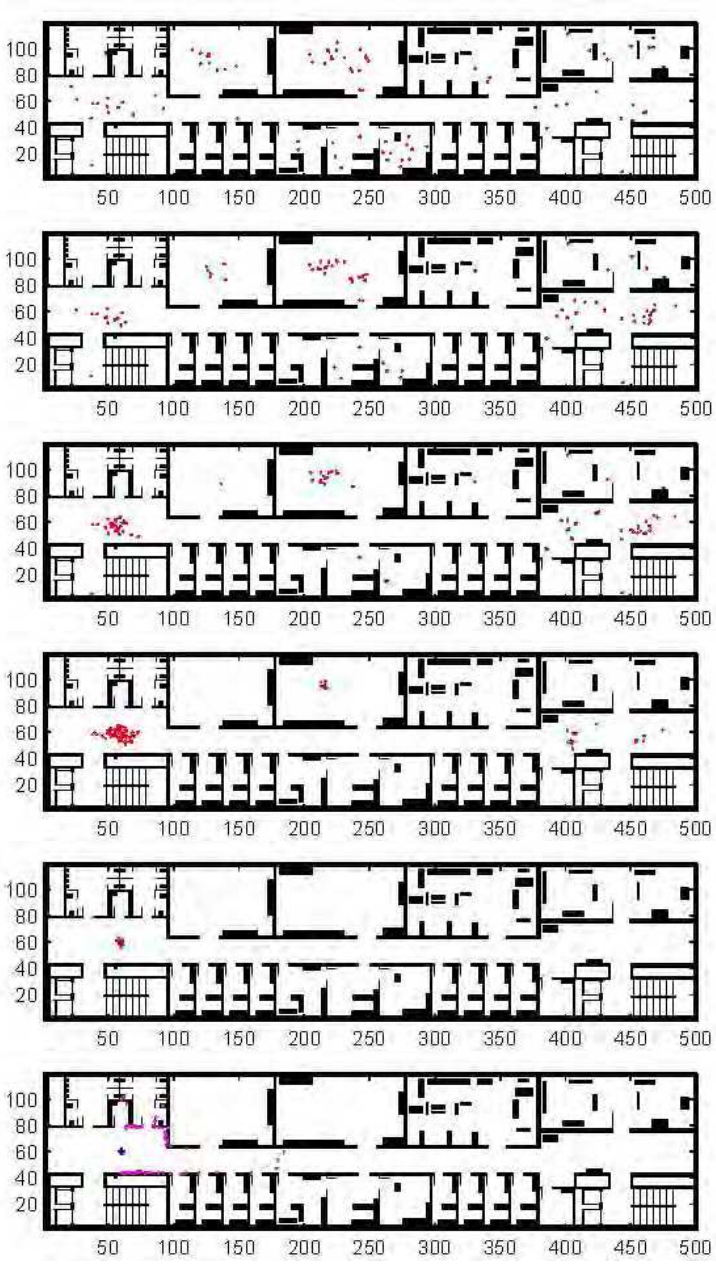


Fig. 5. Convergence process along different iterations (50, 100, 150, 200, 250, 334**)** of the first perception cycle.

6.2. Test 2

The second test tries to determine the capability of the algorithm to identify all possible feasible areas in case the robot is located in a non-distinguishable pose. The true robot position is $(x,y,\theta)^T = (60,60,0)^T$ and the variance considered for each measurement in the simulation is of 3% of the measured signal, which is relatively noisy compared with real laser scanners. The population set used in the test is of 100 elements. Figure 6 shows the convergence population process along the iterations of the first perception cycle. It can be noticed how the algorithm is able to concentrate the population in the most feasible areas (in this case, in offices not localized in front of a laboratory door). The most interesting aspect of this second test is that the algorithm does not eliminate any potential hypothesis.

After the first perception cycle the robot is moved upward (y direction) in the map: 2.5 cells at each motion plus a Gaussian error. After the motion, the whole population set is moved according to the odometry information and a new localization cycle is started, this time with an amplification factor $F = 0.05$. Figure 7 shows the successive population convergence toward an only hypothesis.

In the test example of figure 7, the stopping condition changes following the next sequence: 314, 24, 13, 11, 68 and 9. It can be noticed how the algorithm is heavier in the first perception cycle since it needs to eliminate infeasible areas which require a high number of pose trails. After that iteration, the stopping criteria is reached faster, requiring a number of iterations of two orders of magnitude to converge. It can also be noticed that the number of iterations increases when the robot goes out of the office and perceives the corridor. In that case, the algorithm requires 68 iterations before reaching the stopping criteria. Once the robot observes the corridor, it is able to converge to only one feasible pose area, since the observations that let the algorithm disambiguate between the offices.

6.3. Test 3

The third test tries to determine the capability of the algorithm to identify all possible feasible areas in case the robot is localized at the worst possible case. The worst case happens when the robot is localized at a corner and observes the corner from a short distance. In that case, it is a non-distinguishable pose and the number of possible feasible poses exploits. The true robot's position is $(x,y,\theta)^T = (10,70,135)^T$, which corresponds to the upper left corner of the hall localized at the left side of the test environment. The variance considered for each measurement in the simulation is of 1% of the measured signal. Due to the fact that the number of potential feasible poses is high, if a normal population is used (we understand by normal population a population able to localize the vehicle in normal situations), the algorithm fails because it does not have enough elements to manage the set of potential hypotheses. In our experimental test, a minimum number of approximately $15-25$ elements per potential pose is required. A population set of 1500 of elements has been used to manage properly the high number of feasible hypotheses existent at the initial cycle of the global localization. Figure 8 shows the high number of feasible poses existent at the initial robot's pose according to the perceived information. The number of feasible poses at the end of the first perception cycle is of 54. In this example, a population of is enough, but at the end of the first cycle it has feasible poses. If we decrease the population, the number of manageable poses decreases and the risk of incorrect convergence increases.

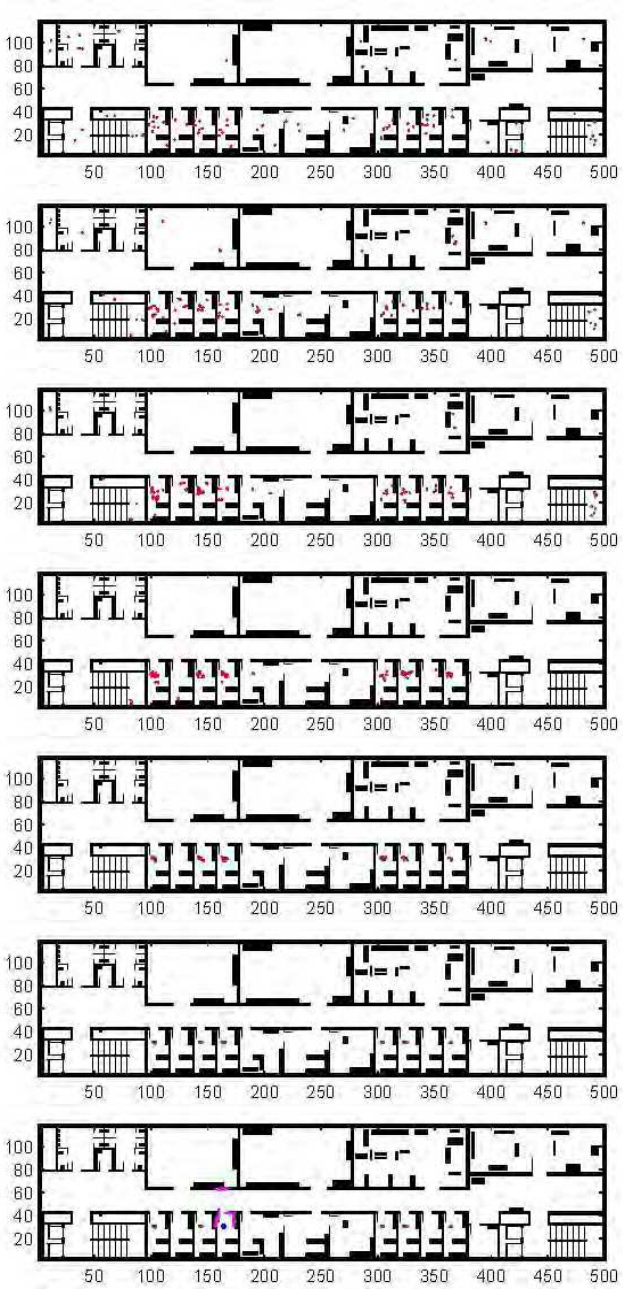


Fig. 6. Convergence process along different iterations (50, 100, 150, 200, 250, 300, 314**)** of the first perception cycle.

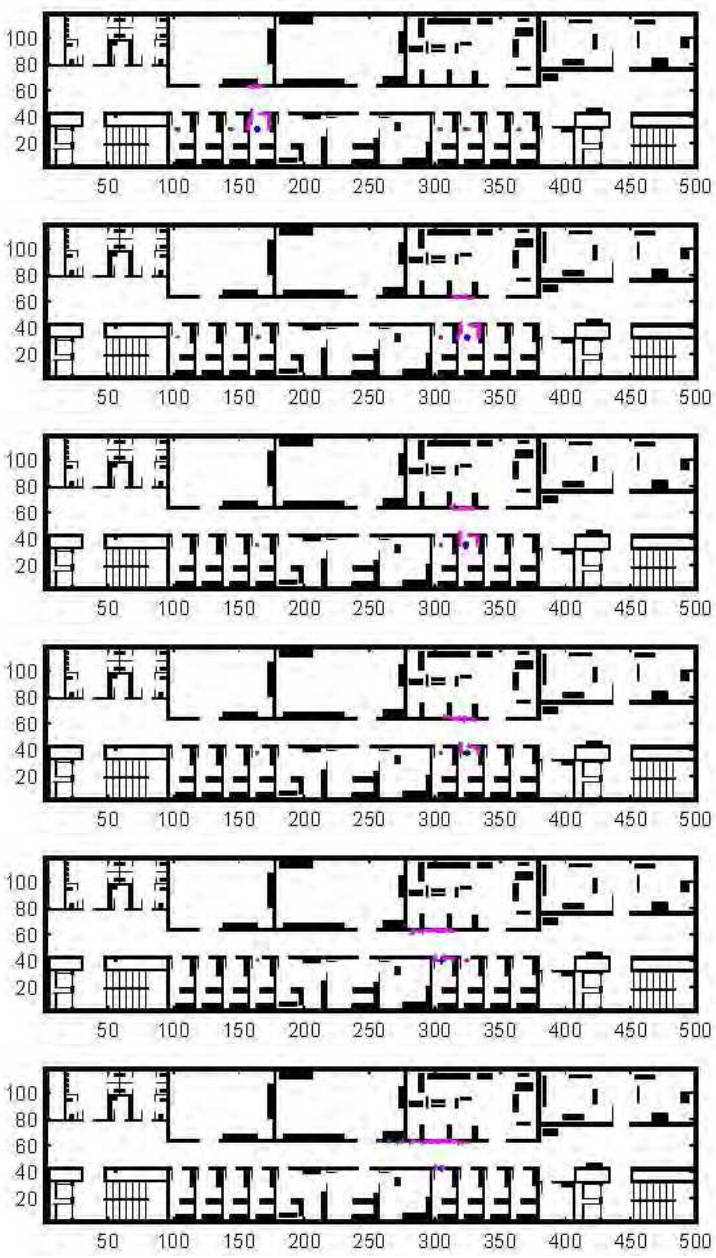


Fig. 7. Convergence process along 6 successive perception-motion cycles where the robot moves 2.5 cells upward after each perception cycle.

After the first perception cycle, the robot is turned 10 degrees clockwise at each motion (plus a random error added and unknown for the algorithm). After the motion, the whole population set is moved according to the odometry information and a new localization cycle is started. Figure 9 shows the successive population convergence towards an unique hypothesis. After the initial perception cycle, the number of possible poses is pruned very fast since new information about the environment is added to the algorithm.

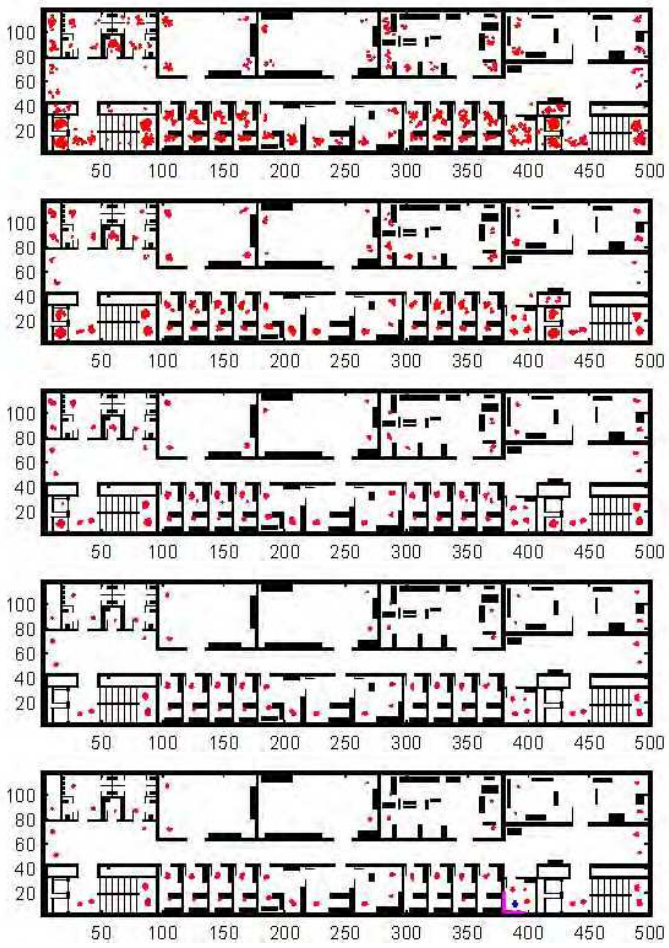



Fig. 8. Convergence process for the worst case pose (100, 200, 300, 400, 420) of the first perception cycle.

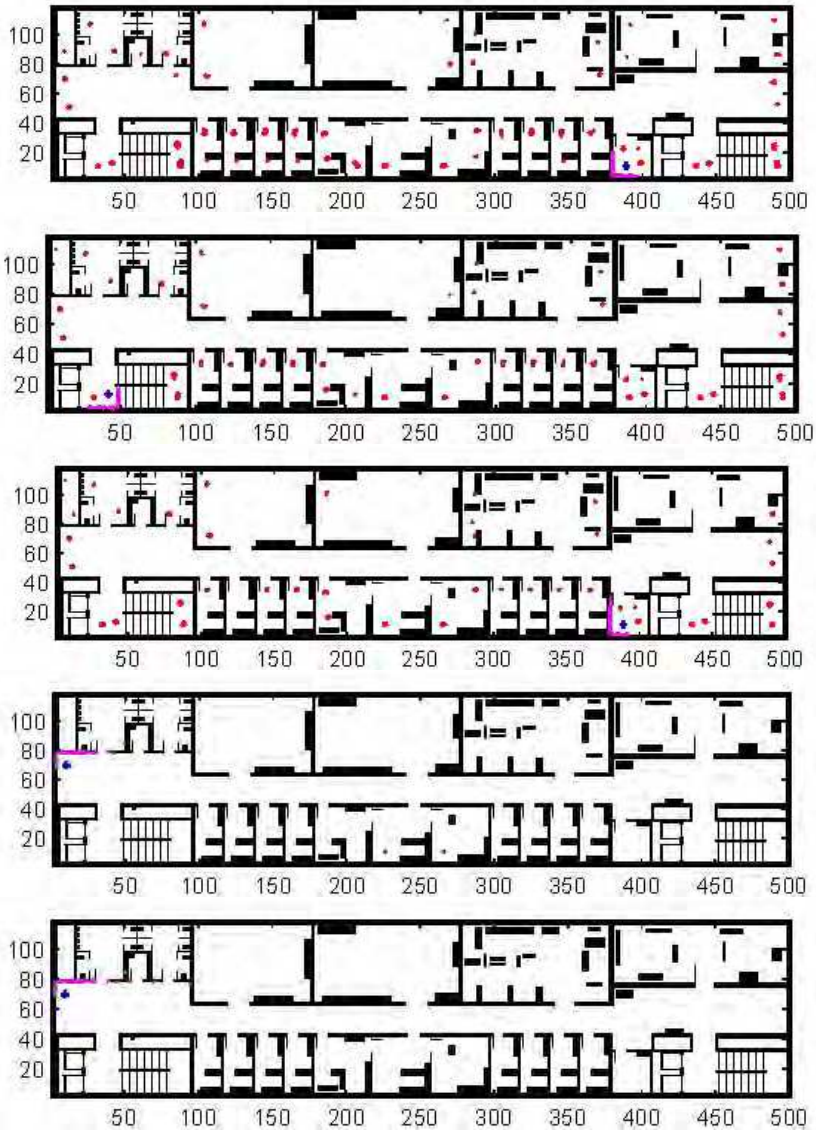



Fig. 9. Convergence process along 5 successive perception-motion cycles where the robot turns 10° clockwise after each perception cycle.

From a practical point of view, the worse case situation is easy to alleviate by turning the vehicle up to a pose with the maximum perception information. For this same example, if we start the localization with an initial pose of $(x, y, \theta)^T = (10, 70, 135)^T$ where the perceived area covered by robot sensors is maximum, the problem disappears and a population of 100 elements is enough, as can be noticed in figure 10.

7. Accuracy results

To test the accuracy of the method for the initial localization, we have selected again a distinguishable point (x,y,θ)=(60,60,0) and we have increased the variance of the error in sensor measurement. The variance considered in simulations is proportional to the measured signal and is expressed as a fraction of the total measurement. This way, a value of 0.01 indicates a variance of 1 % over the simulated value obtained for a given sensor. This situation is harder than real conditions in laser scanners. The population used in simulation is of 100 elements. For each case, 25 runs of the first perception cycle have been executed. The results are shown in table 1, where the mean and variance of the absolute errors in x, y and θ are given. It also shows the average number of iterations required by the algorithm until the stopping criteria is reached and the success ratio obtained for each noise level.

It can be notice that, for low noise variance levels (up to 5%), the accuracy of the algorithm is below 0.15 cells in x and y dimensions and below 0.2° in orientation in all the cases. Since the cell size used is 12 cm, that means an error below 1.8 cm in x and y dimensions and below 0.2° in orientation at the end of the first perception cycle. For this signal error level, the algorithm has successfully localized the true pose in all the runs and only one hypothesis is maintained at the end of the first perception cycle. The stopping criteria is reached in a relatively constant number of iterations for low noise levels, and it tends to decrease slowly when the noise signal level increases.

The algorithm degrades slowly. For a 17.5% of variance in the noise level, the algorithm is able to localize a position close to the true one in all simulations. We consider a localization as successful if the initial pose estimated is in a 10 cells area around the true one. After that level, the success ratio drop fast and, for a 25% of variance in the noise level, the success ratio value is only of a 60%.

From this test, some conclusions can be drawn. The first one is that, if the place is distinguishable, the algorithm is able to localize the initial pose with high accuracy, and only when the noise increases considerably the algorithm starts to decrease its success ratio.

A second aspect to consider is the capability of the algorithm to maintain bounded the pose estimation accuracy along a trajectory. A motion along the central corridor in the test environment is simulated. For the simulations, a normal error with 2.5% of variance has been adopted and the noise used for sensor observation is of 1% of variance. The real and estimated trajectories are shown in figure 11 and the x, y, and θ errors are shown in figure 12. The simulation results show that y error is very small and contained between [+0.05, -0.05] cells ([+0.6, -0.6] cm). This is logical, since the corridor is relatively narrow and the y position can be accurately estimated with the available information. For x variable, errors are in a band between [+0.1, -0.1] cells ([+1.2 -1.2] cm) and they punctually reach values in the band [+0.25, -0.25] cells. Regarding the orientation, θ errors are in a band between [+0.1, -0.1]° and they punctually reach values in the band [+0.25, -0.25]°. The error goes to -0.31° in one occasion.

$\sigma_{\mathbf{m}}$	$\lvert e \rvert_x$	σ_x	$\lvert e \rvert_y$	σ_y	$\lvert e \rvert_\theta$	σ_{theta}	*It*	*Suc*
0.01	0.051	0.052	0.021	0.017	0.046	0.044	295.8	1.0
0.02	0.112	0.091	0.031	0.031	0.082	0.058	342.6	1.0
0.03	0.140	0.125	0.040	0.036	0.081	0.081	362.5	1.0
0.04	0.134	0.133	0.067	0.047	0.120	0.071	365.7	1.0
0.05	0.139	0.098	0.132	0.147	0.180	0.188	298.5	1.0
0.075	0.276	0.248	0.212	0.266	0.408	0.588	316.2	1.0
0.10	0.416	0.383	0.301	0.277	0.485	0.525	298.6	1.0
0.125	0.374	0.308	0.529	0.415	0.835	0.765	246.7	1.0
0.15	1.255	2.478	0.816	0.506	1.420	1.275	291.9	1.0
0.175	0.598	0.573	0.844	0.603	1.369	0.863	305.4	1.0
0.20	1.056	0.683	0.9611	0.705	2.186	1.641	294.4	0.96
0.225	2.242	3.426	1.5681	1.365	2.457	1.891	288.9	0.68
0.25	3.069	2.575	1.5849	1.252	1.826	1.384	264.3	0.60

Table 1. Accuracy of the algorithm for different levels of noise, true location (60,60,0).

7.1. Computational cost

When analyzing the computational cost of the algorithm, two different situations have to be considered:

• *Initial localization cycle.* In this situation, the algorithm explores the full state space until the stopping condition is reached. The time used to reach the stopping condition depends on the worst pose value considered as threshold in the stopping criteria. This time for the test example is around 4.5 seconds (in a T8300 duo core processor at 2.4 GHz with one core at execution). This time depends on the population set, the stopping criteria and the sensed area. The sensor perception estimation is done by ray tracing on the environment map and the estimation cost tends to grow with the size of the observed area since the algorithm concentrates its exploration in feasible areas. At the end of this first perception cycle, the feasible localization areas are determined.

• *Re-localization cycle*. Once the algorithm detects that the whole initial population has converged, the population set is decreased to 30 elements. For this population set the stopping condition is reached very fast and the computational cost decreases to 45 milliseconds per iteration.

These times allow the algorithm to be used on line except at the initial cycle.

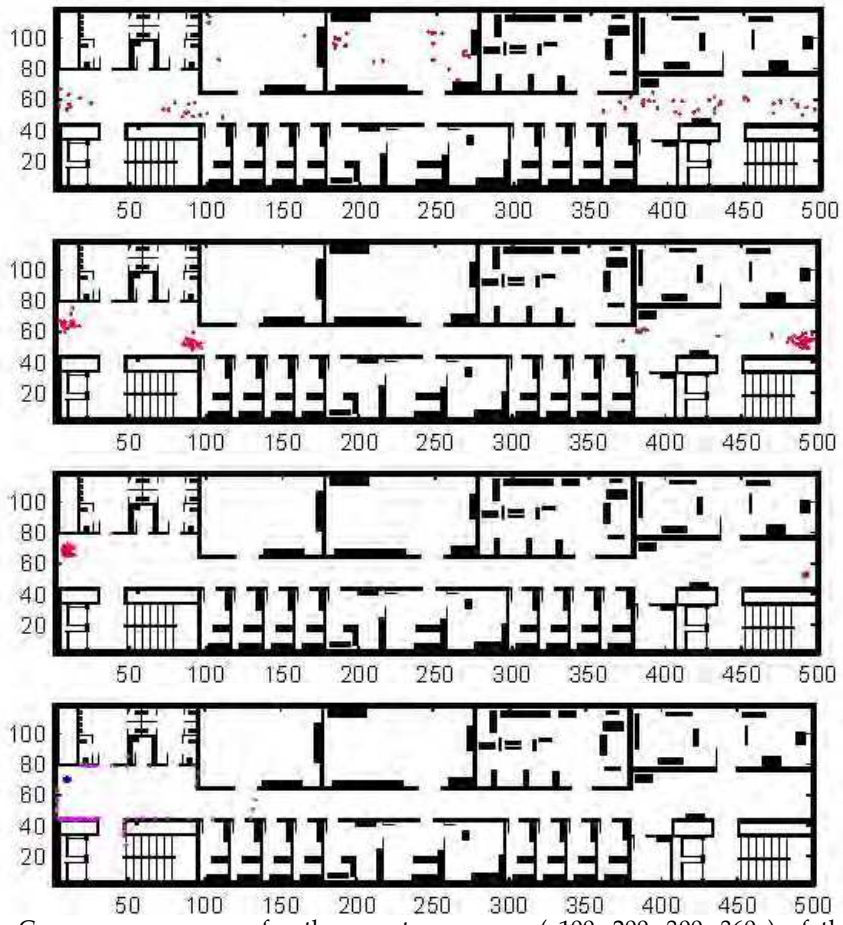


Fig. 10. Convergence process for the worst case pose (100, 200, 300, 360) of the first perception cycle.

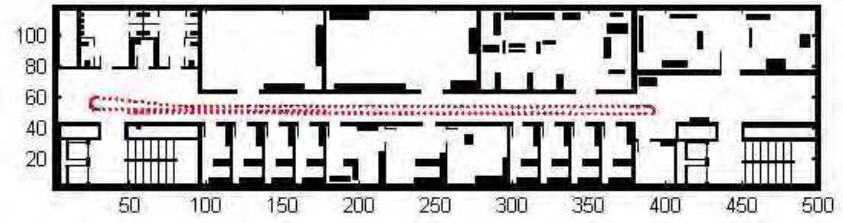


Fig. 11. Real and estimated trajectories along the central corridor.

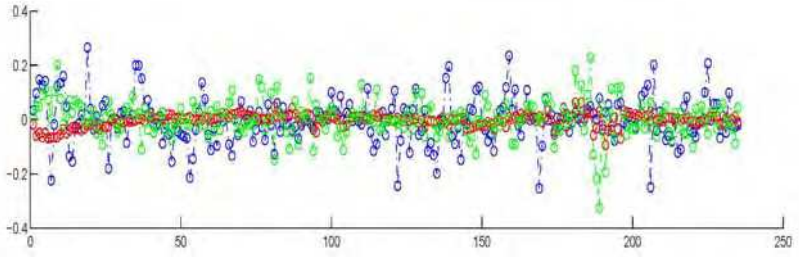


Fig. 12. Errors of the trajectory of Fig. 11 (in cells in x and y, and in degrees in θ).

8. Conclusion

One of the most interesting properties of the RDE (Rejection Differential Evolution) filter is its efficiency in terms of convergence speed. The algorithm is able to exploit the perceptual information to converge in the initial perception cycle if, from the initial pose, the robot perceives any distinctive characteristic of its environment. A second characteristic is the low number of pose solutions required to successfully localize the robot. For the test environment under consideration, a population of 100-150 elements is enough, except in non-informative situations where the number of hypotheses can grow up very fast and a certain number of pose solutions is required to maintain a feasible area.

The number of population elements required to avoid the premature elimination of feasible hypotheses has not been determined theoretically in the evolutive algorithms field, but in our experimental tests, a number between 10 and 25 poses is required to maintain all feasible hypothesis. In case of non informative situations where the sensors only let the robot perceive a small part of the environment (for instance, when a robot is in a corner) the potential number of hypotheses can rise very fast, which originates that the algorithm fails when using a normal pose set size. This problem can be addressed in two ways: turning the robot until a maximum environment area is perceived by the sensors or to detect the uninformative situation and increase the pose set size. The first approach is easier and requires less computational resources.

As in the majority of the population-based optimization methods, the algorithm robustness increases with the population set size. If we consider the effect of the population size on the accuracy of the algorithm, we need to consider the explored number of poses, since the total number of explored poses is roughly speaking the product of the iteration number and the population size. But in our test, the accuracy is maintained up to a certain number of explored poses. This behavior differs completely from Monte Carlo method. 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, places not enough samples in regions where the desired posterior probability distribution is large. This problem has practical importance because of time limitations existing in on-line applications.

The use of a rejection threshold and a stopping criteria adjusted to the statistical characteristics of the objective function allows us to decrease considerably the population size while maintaining the accuracy level of the method. In previous works, a minimum population set of 250 – 300 elements were required, while in the RDE version a population

of 100 has proved to be sufficient for the environment under consideration. At initial stages, the algorithm has to evaluate the whole environment map and the initial number of samples is related to the environment area and the area perceived with sensors. If the perceived area is big, the possible number of hypotheses required for the environment can be reduced and, consequently, the population required.
A significant characteristic of the method is the possibility of stopping the algorithm convergence to avoid the premature elimination of feasible hypotheses until posterior motion-perception cycles can add a significant statistical evidence to achieve the convergence to one pose. This characteristic is critical when the initial robot's pose is localized in a repetitive place, such as an office, without singular characteristics and the algorithm needs to detect the feasible hypotheses and to maintain them until distinctive characteristics are perceived in posterior perception cycles.
Due to the stochastic nature of the algorithm search of the best robot's pose estimate, the algorithm is able to cope with 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.

9. References

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

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

Branke, J.; Schmidt, C. & Schmeck, H. (2001). Efficient fitness estimation in noisy environments, In: *Proc. of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pp 243-250, Morgan Kaufmann

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

Cox, I. J. (1991). Blanche-An Experiment in guidance and navigation of an autonomous robot vehicle, In: *IEEE Transactions on Robotics and Automation*, **7**, pp 193-204

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

Crowley, J. L. (1989). World modelling and position estimation for a mobile robot using ultrasonic ranging, In: *IEEE International Conference on Robotics and Automation*, Scottsdale, AZ

Dellaert, F.; Fox, D.; Burgard, W.; & Thrun, S. (1999). Monte Carlo Localization for Mobile Robots, In: *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.; & Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, **11**, pp 391-427

Garrido, S.; Moreno, L. & Salichs, M.A. (1998). Non linear on line identification of dynamic systems with restricted genetic optimization, In: *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing, EUFIT*, pp. 423-428

Garrido, S. & Moreno, L. (2001). Learning adaptive Parameters with Restricted Genetic Optimization, In: *Bio-inspired Applications of Connectionism: 6th International Work-Conference on Artifical and Natural Neural Networks, IWANN2001*, pp 612-620, Springer-Verlag

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

He, J. & Kang, L. (1999). On the convergence rates of genetic algorithms. *Theoretical Computer Science*, 229, pp 23-39

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

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

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

Kalos, M. H. & Whitlock, P.A. (1986). *Monte Carlo Methods. Volume I: Basics*, John Wiley and Sons

Krink, T.; Filipic, B.; Fogel, G. & Thomsen, R. (2004). Noisy Optimization Problems - A Particular Challenge for Differential Evolution, In: *Proc. of the 2004 IEEE Congress on Evolutionary Computation*, pp 332-339, IEEE Press

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

Leonard, J.J. & Durrant-White H.F. (1992). *Directed sonar sensing for mobile robot navigation*. Kluwer Academic Publishers

Lenser, S. & Veloso, M. (2000). Sensor resetting localization for poorly modelled mobile robots, In: *Proc. IEEE International Conference on Robotics and Automation (ICRA-2000)*, San Francisco, C.A.

Markon, S., Arnold, D. V.; Baeck, T.; Bielstein, T. & Beyer, H.- G. (2001). Thresholding- a selection operator for noisy ES., In: *Proc. of the 2001 IEEE Congress on Evolutionary Computation*, pp 465-472, IEEE Press

Moreno, L.; Garrido, S. & Muñoz, M. L. (2006). Evolutionary filter for robust mobile robot localization, In: *Robotics and Autonomous Systems*, 54, pp 590-600

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

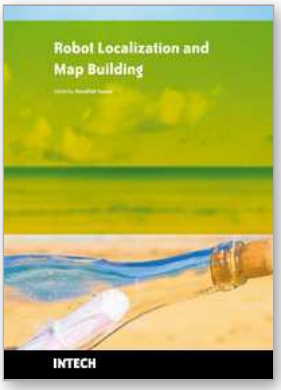
Roumeliotis, S. I. & Bekey, G.A. (2000). Bayesian estimation and Kalman filtering: A unified framework for mobile robot localization, In: *Proc. IEEE International Conference on Robotics and Automation (ICRA-2000)* , pp 2985-2992, San Francisco

Storn, R. & Price, K. (1995). Differential Evolution- A simple and efficient adaptive scheme for global optimization over continuous spaces. *Technical Report TR-95-012*, March

Thrun, S. (1998). Bayesian landmark learning for mobile robot localization, *Machine Learning*, 33(1)

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

Vahdat, A. R.; Ashrafoddin, N. N. & Ghidary, S.S. (2007). Mobile Robot Global Localization using Differential Evolution and Particle Swarm Optimization, In: *Proc. IEEE Congress on Evolutionary Computation (CEC-2007)*, pp 1527-1534



Robot Localization and Map Building

Edited by Hanafiah Yussof

ISBN 978-953-7619-83-1
Hard cover, 578 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

Localization and mapping are the essence of successful navigation in mobile platform technology. Localization is a fundamental task in order to achieve high levels of autonomy in robot navigation and robustness in vehicle positioning. Robot localization and mapping is commonly related to cartography, combining science, technique and computation to build a trajectory map that reality can be modelled in ways that communicate spatial information effectively. This book describes comprehensive introduction, theories and applications related to localization, positioning and map building in mobile robot and autonomous vehicle platforms. It is organized in twenty seven chapters. Each chapter is rich with different degrees of details and approaches, supported by unique and actual resources that make it possible for readers to explore and learn the up to date knowledge in robot navigation technology. Understanding the theory and principles described in this book requires a multidisciplinary background of robotics, nonlinear system, sensor network, network engineering, computer science, physics, etc.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

M.L. Munoz, L. Moreno, D. Blanco and S. Garrido (2010). Global Localization Based on a Rejection Differential Evolution Filter, Robot Localization and Map Building, Hanafiah Yussof (Ed.), ISBN: 978-953-7619-83-1, InTech, Available from: http://www.intechopen.com/books/robot-localization-and-map-building/global-localization-based-on-a-rejection-differential-evolution-filter

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2010 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.