Robotics and Autonomous Systems 59 (2011) 597–610

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

L1–L2-norm comparison in global localization of mobile robots

L. Moreno [a], D. Blanco [a], M.L. Muñoz [b], S. Garrido [a,*]

[a] *Robotics Lab, Carlos III University of Madrid, Madrid, Spain*

[b] *Faculty of Computer Science, Polytechnic University of Madrid, Madrid, Spain*

A R T I C L E   I N F O

Article history:
Received 1 June 2010
Received in revised form
5 April 2011
Accepted 19 April 2011
Available online 7 May 2011

Keywords:
L1-norm
L2-norm
Differential evolution
Nonlinear filter
Global localization
Mobile robots

A B S T R A C T

The global localization methods deal with the estimation of the pose of a mobile robot assuming no prior state information about the pose and a complete a priori knowledge of the environment where the mobile robot is going to be localized. Most existing algorithms are based on the minimization of an L2-norm loss function. In spite of the extended use of the L2-norm, the use of the L1-norm offers some alternative advantages. The present work compares the L1-norm and the L2-norm with the same basic optimization mechanism to determine the advantages of each norm when applied to the global localization problem.

The algorithm has been tested subject to different noise levels to demonstrate the accuracy, effectiveness, robustness, and computational efficiency of both L1-norm and L2-norm approaches.

1. Introduction

The statistical methods are based on assumptions which formalize the information known or conjectured about the data of the problem. The most popular model formalization is the assumption that the observed data have a Gaussian distribution. This assumption has been the framework for the classical least squares estimation method, which has been the dominating technique in the engineering literature for a long time. There are two main reasons for assuming a normal distribution: on the one hand, it gives a good approximate representation to many real data observation sets, and on the other hand, it is theoretically quite convenient because the continuity of the loss function obtained from it allows us to derive explicit formulas for optimal statistical methods such as the maximum likelihood estimators, the least squares method, or the Kalman-based estimators. Such methods rely on the assumption that a normal distribution is exactly followed. In spite of these well-known characteristics, the least squares methods are far from the optimal in many non-Gaussian situations, particularly when the error distributions have long tails. In these situations, any outlying observation may affect the method seriously. A traditional way of avoiding this problem in engineering is the use of the Chi-squared test and the Mahalanobis distance to reject those measurements with low probabilities. This alleviates the problem but requires the noise to be Gaussian and, in certain situations, it can reject useful information.

When the noise model is not exactly known or the model is contaminated with another probability distribution, the least squares method is not completely satisfactory. In the present work, we are interested in obtaining high accuracy in the estimation of the robot's pose. The quadratic loss function (L2-norm) tends to give more importance to big errors than to small ones, which originates a limited accuracy.

The use of the absolute error (L1-norm) may be a more satisfactory measure of loss than the squared error in certain situations. The minimum sum of absolute errors overcomes the aforementioned drawbacks of the least squares method and provides an alternative to be explored. It is less sensitive than the least squares regression to the high errors originated in the outliers or in the case of contamination. It may be noted that the absolute error estimates are of maximum likelihood and hence asymptotically efficient when the errors follow the Laplace distribution. The minimum sum of the absolute error regression has been studied in different contexts and has received different names: minimum sum of absolute errors (MSAE) [1], least sum of absolute errors (LSAE) [2], minimum absolute deviation errors (MAD) [3], least absolute deviation errors (LAD) [3], L1-norm, etc. It has been successfully used in different fields. Based on Monte Carlo studies, the use of the L1-norm has been recommended whenever the errors follow a Laplace or a Cauchy distribution [4], a mixture of normal and uniform distributions [5], and a contaminated normal distribution [6]. The L1-norm estimates have been strongly penalized by its computational

* Corresponding author. Tel.: +34 91 624 6012; fax: +34 91 624 9430.
E-mail addresses: moreno@ing.uc3m.es (L. Moreno), sgarrido@ing.uc3m.es (S. Garrido).

burden. Since the L1-norm function is not derivable, the optimization problem leads to linear programming methods to obtain a solution. Dielman [7] showed, using a Monte Carlo simulation, that when outliers are present, the least absolute value forecasts are superior to the least squares forecasts.

The use of L1-norms introduces problems related to the estimation time. Unlike L2-norm estimators, no analytic expression can be formulated for the estimator. The most classical method to obtain the estimator is by applying a linear programming algorithm to the data. This difficulty to obtain a closed expression for the estimator makes it difficult to determine its properties analytically and it needs to be demonstrated by sampling.

The global localization problem applied to mobile robots tries to estimate the initial robot's pose assuming that the robot has no knowledge about its initial position. Therefore, it has to estimate the robot's pose globally. To solve the global localization problem there exist different approaches: *Bayesian-based estimation methods*, *optimization-based methods*, and *hybrid methods*.

In *Bayesian-based methods*, all existing probabilistic information (sensor and motion information) is integrated 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. These approaches do not use any loss function and do not require any loss function derivation to obtain a solution, at least in the probability density integration process. However, to obtain the maximum a posteriori point they require a way of obtaining the maxima, which can require a derivative of the a posteriori probability density function. In practice, most of them are Monte Carlo variations, which alleviates the probability density integration problem and the search for the probability distribution maxima. These methods can manage multi-hypotheses or multi-minima problems easily, but are computationally intensive. Among the purely Bayesian methods, we can include grid-based probabilistic filters [8–10] and Monte Carlo localization methods [11–13].

Optimization-based methods also use all existing probabilistic information to obtain a loss function that is minimized at each motion–perception cycle and the estimate is the point with lower value of the loss function at a given moment. Among these methods, we can differentiate between two groups: those based on the derivation of the loss function to achieve a solution and those based on the stochastic search for the best solution of the loss function. Among the derivative-based optimization methods, we can include the Kalman filters. The drawback of these methods is their inability to deal with multi-hypotheses problems; their main advantage is their computational speed. They are widely used in tracking problems (re-localization), where the method only requires to manage one hypothesis. Among the optimization-based approaches that do not use derivatives we find: Moreno [14] presents an evolutive localization algorithm known as Evolutive Localization Filter (ELF), Martin [15] introduces the ELF-3D algorithm that uses the three dimensional sensorial information, differential evolution Monte Carlo [16] and particle swarm optimization filters [17].

Hybrid methods combine both classes of methods. Multi-hypothesis Kalman filters are a combination of Bayesian methods and optimization-based methods. They maintain a set of multi-hypotheses, each of them with an associated Gaussian probability [18–22]. Each hypothesis is managed by a Kalman filter method and most of them use a decision tree search mechanism based on geometrical constraints together with probabilistic attributes to manage the global data association problem.

Except for the Monte Carlo methods, where no loss function is minimized, the rest of methods use the L2-norm as the loss function to minimize. This paper presents a comparison between the solution to the global localization problem obtained with an L1-norm and an L2-norm. In the present work the method used to solve the global localization problem is based on an improved version of the Evolutionary Localization Filter method called Adaptive Evolutionary Localization Filter (A-ELF) [14]. The A-ELF solves the global localization problem in two phases. In the first phase, the algorithm searches globally to find the feasible areas and lets the whole set of possible solutions interact to extensively explore the state space in order to make that the set of tentative solutions converges to the set of most probable areas (given the loss function definition). In a second phase, the amplification factor is decreased considerably to focus the algorithm exploration around the selected areas. The algorithm operates with raw sensor data, avoiding the extraction of features from the sensor data, and uses the loss function adopted in the previous step to estimate the new set of poses. The use of a probabilistic fitness function instead of estimating the probability density function offers the possibility of iterating as many times as necessary over the data until certain pre-established statistic conditions are reached. This can be done because the fitness associated to a solution indicates the quality score of the solution, given the data and the robot's motion.

2. Localization problem formulation

From a probabilistic point of view, the global localization problem searches to estimate the pose which maximizes the a posteriori probability density. This problem consists of two linked problem. On the 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 measurements, and the a priori environment map. On the other hand, an optimization problem to determine the point $\hat{x}_t^{\mathrm{MAP}}$ with maximum a posteriori probability density at a given time. Depending on the method adopted, the solution is obtained by focusing on the integration of the probabilistic information or on the optimization of a loss function.

The robot's pose at time t will be denoted by $x_t = (x, y, \theta)^T$, and the data up to time t by Y_t. The posterior probability distribution can be written as $p(x_t|Y_t)$, where the environment model is assumed to be known. The sensor data come from two different sources: *motion sensors* which provide data related to pose changes $u(t-1)$ originated by the robot's displacement in the time interval $[t-1, t]$ (e.g., odometer readings), and *perception sensors* which provide data related to environment z_i (e.g., camera images, laser range scans, ultrasound measurements). We will consider that both types of data arrive alternatively, $Y_t = \{z_0, u_0, \dots, z_{t-1}, u_{t-1}, z_t\}$. These sensor data can be divided into two groups of data $Y_t \equiv \{Z_t, U_{t-1}\}$, where $Z_t = \{z_0, \dots, z_t\}$ contains the perception sensor measurements and $U_{t-1} = \{u_0, \dots, u_{t-1}\}$ contains the odometry information. To estimate the posterior distribution $p(x_t|Y_t)$, the probabilistic approaches resort to the *Markov assumption*, which states that the future states only depend on the knowledge of the current state and not on how the robot got there, i.e., they are independent of past states.

$$\begin{aligned}\hat{x}_t^{\mathrm{MAP}} &= \arg\max_x p(x_t|Y_t)\\ &= \arg\max_x \prod_{i=1}^{t} p_e(z_i|x_i) \prod_{i=1}^{t} p_v(x_i|x_{i-1}, u_{t-1}) p(x_0). \qquad (1)\end{aligned}$$

This expression requires to specify $p_v(x_t|x_{t-1}, u_{t-1})$, $p_e(z_t|x_t)$, and $p(x_0)$, where $p_e(z_t|x_t)$ expresses the probability density function for observation z_t, given the state x_t for the observation noise e; and $p_v(x_t|x_{t-1}, u_{t-1})$ indicates the probability density function for the motion noise v. The expression to be maximized can be reformulated in an equivalent and more convenient form by taking logarithms and expressing the resulting objective function $f_0(t)$ recursively:

$$f_0(x_t) = \sum_{i=1}^{t} \log p_e(z_i|x_i) + \sum_{i=1}^{t} \log p_v(x_i|x_{i-1}, u_t) + \log p(x_0)$$
$$= \log p_e(z_t|x_t) + \log p_v(x_t|x_{t-1}, u_{t-1}) + f_0(t-1). \quad (2)$$

If we are able to solve the optimization problem at time $t-1$ and we have a set of sub-optimal solutions x^*_{t-1} which are candidates to satisfy the optimization problem up to time $t-1$, the optimization problem can be reformulated as

$$\hat{x}_{t-1} = \max_x(\log p_e(z_t|x_t) + \log p_v(x_t|x_{t-1}, u_{t-1})) \quad (3)$$

where $\hat{x}_{t-1}$ is the x that solves the maximum a posteriori optimization problem at time $t-1$. Then, solving Eq. (3) we will obtain a recursive version of the MAP estimate. Up to this point, no assumption about the noise distribution has been necessary. In the following sections, p_e and p_v will be written as p to alleviate the notation.

3. Loss function derivation

According to the optimization problem in Eq. (3), the natural choice for the loss function is

$$f_0(x_t) = \log p(z_t|x_t) + \log p(x_t|x_{t-1}, u_{t-1}). \quad (4)$$

This expression contains the perception error probability density distribution $p(z_t|x_t)$ and the robot's motion error probability density distribution $p(x_t|x_{t-1}, u_{t-1})$. A third probability model is used to model the information we have at the initial stage about the initial a priori robot's pose $p(x_0)$. In case of global localization problem, the initial pose information is null. Then, a uniform probability distribution along the state space is assumed.

3.1. L2-norm

If we assume that the observation error can be described by a Gaussian probability distribution with zero mean and known variance, then the probability of observing $z_{t,i}$ with sensor i can be expressed as

$$p(z_{t,i}|\hat{x}_t) = \frac{1}{(2\pi\sigma_e^2)^{1/2}} e^{-1/2\frac{(z_{t,i}-\hat{z}_{t,i})^2}{\sigma_e^2}}. \quad (5)$$

Integrating all the individual sensor beam probabilities into a joint probability value and assuming conditional independence between the individual measurements result in

$$p(z_t|\hat{x}_t) = \prod_{i=0}^{N_s} p(z_{t,i}|\hat{x}_t) = \prod_{i=0}^{N_s} \frac{1}{(2\pi\sigma_e^2)^{1/2}} e^{-1/2\frac{(z_{t,i}-\hat{z}_{t,i})^2}{\sigma_e^2}} \quad (6)$$

where N_s is the number of sensor observations.

Assuming that the motion error is a Gaussian probability distribution with zero mean and known variance, that is, $v \approx N(0, P)$, then the motion error probability $p(x_i|x_{i-1}, u_{i-1})$ can be expressed as

$$p(x_i|x_{i-1}, u_{i-1}) = \frac{1}{\sqrt{|P|(2\pi)^n}} e^{-1/2(x_i-\hat{x}_i)P^{-1}(x_i-\hat{x}_i)^T}. \quad (7)$$

Introducing the expressions of $p(x_t|x_{t-1}, u_{t-1})$ and $p(z_t|x_t)$ in Eq. (4), we have

$$f_0(x_t) = \sum_{i=0}^{N_s} \log(2\pi\sigma_e^2)^{-1/2} - \sum_{i=0}^{N_s} \frac{(z_{t,i}-\hat{z}_{t,i})^2}{2\sigma_e^2}$$
$$+ \log[(|P|(2\pi)^n)^{-1/2}] - \frac{1}{2}(x_i-\hat{x}_i)P^{-1}(x_i-\hat{x}_i)^T. \quad (8)$$

The constant terms can be eliminated and the problem is reduced to find the robot's pose which minimizes

$$f_0'(x_t) = \sum_{i=0}^{N_s} \frac{(z_{t,i}-\hat{z}_{t,i})^2}{2\sigma_e^2} + \frac{1}{2}(x_i-\hat{x}_i)P^{-1}(x_i-\hat{x}_i)^T. \quad (9)$$

This classical derivation of the loss function leads to an L2-norm loss function $N(\mu, \sigma^2)$.

3.2. L1-norm

In practice, the assumption of a Gaussian observation noise is arguable. On the one hand, the presence of non-modeled obstacles, both static and mobile, lets us notice that the Gaussian probability distribution can be convenient, but in practice the distribution tail is too optimistic. If we assume that the observation error can be described by a Laplace error distribution, this distribution has the following form

$$f(x|\mu, \lambda) = \frac{1}{2\lambda} \exp\left(-\frac{|x-\mu|}{\lambda}\right) \quad (10)$$

with mean μ and variance $2\lambda^2$, usually referred to as location and scale parameters, respectively, and typically denoted as $L(\mu, \lambda)$. If we apply this probability distribution to model the observation and motion noise, we obtain an L1-norm loss function to optimize

$$f_0''(x_t) = \sum_{i=0}^{N_s} \left[\frac{|z_{t,i}-\hat{z}_{t,i}|}{\lambda_z} + \frac{|x_i-\hat{x}_i|}{\lambda_x}\right]. \quad (11)$$

In spite of the theoretical interest of a derivation from a Laplace distribution, this distribution can be assumable for the observation error distribution but not so evidently for the motion noise error. However, the problem can be formulated in more general terms and independently of the subjacent error distribution. A very generic expression for a wide class of loss functions is

$$f_0'''(x_t) = \sum_{i=0}^{N_s} \frac{(|z_{t,i}-\hat{z}_{t,i}|^p)^{\frac{1}{p}}}{\sigma_z} + \frac{(|x_i-\hat{x}_i|^p)^{\frac{1}{p}}}{\sigma_x}. \quad (12)$$

This expression includes the L1-norm and the L2-norm as particular cases. The main problem of this loss function derives from the discontinuity introduced by the absolute value function. This eliminates the possibility of obtaining a closed solution even if there is only one minimum. Since the optimization method to be used is not derivative-based, the L1-norm, L2-norm, or any other Lp-norm loss functions can be solved in a completely similar way and the computational cost is not substantially different. In the following section, the nonlinear filter used to obtain the estimates will be introduced.

Eq. (12) defined in this general way is independent of the noise present in the system, which means that one can use the L1-norm or the L2-norm when the noise in the system is Gaussian or when it follows a different distribution. In this work, we will compare both norms under two different perception error situations to observe the advantages and disadvantages of using both of them for global pose estimation.

4. Adaptive Evolutionary Localization Filter (A-ELF) algorithm description

The algorithm proposed to implement the A-ELF is based on the Differential Evolution (DE) method proposed by Storn and Price [23–25] for global optimization problems over continuous spaces. The A-ELF 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

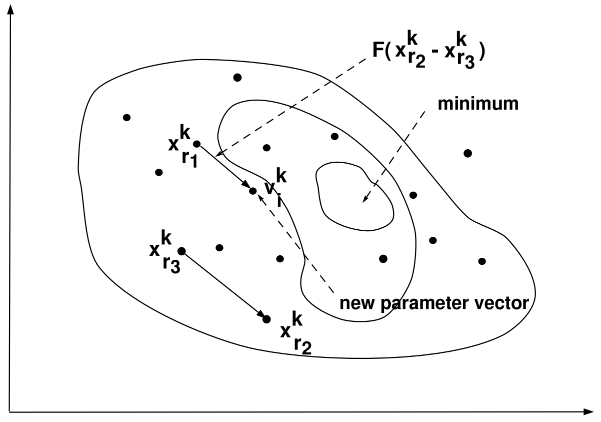


Fig. 1. New population member generation.

DE/rand/1/bin method uses a parallel direct search method which utilizes n-dimensional parameter vectors $x_i^k = (x_{i,1}^k, \dots, x_{i,n}^k)^T$ 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 $\{x_i^k;\ i = 1, \dots, N\}$ as a sub-optimal feasible solutions set (population) for each generation 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. This method uses a differential perturbation method to generate an offspring population. The DE algorithm 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. 1).

4.1. Differential perturbation operation

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

$$v_i^k = x_{r_1}^k + F(x_{r_2}^k - x_{r_3}^k) \tag{13}$$

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. The scaling factor F is a real constant factor which controls the amplification of the differential variation $(x_{r_2}^k - x_{r_3}^k)$.

4.2. Crossover operation

In order to increase the diversity of the new generation of parameter vectors, a crossover operation is introduced. Let us denote by $u_i^k = (u_{i,1}^k, u_{i,2}^k, \dots, u_{i,n}^k)^T$ the new parameter vector generated through the crossover operation between vectors v_i^k and x_i^k, with

$$u_{i,j}^k = \begin{cases} v_{i,j}^k & \text{if } p_{i,j}^k < Cr \\ x_{i,j}^k & \text{otherwise} \end{cases} \tag{14}$$

where each $p_{i,j}^k$ is a randomly chosen value, according to an uniform distribution, from the interval (0, 1) for each parameter j of the population member i at step k. The random values $p_{i,j}^k$ are made anew for each trial vector i. The Cr factor constitutes the crossover control variable and generates a random process with two possible outcomes for each element of vector u. The probability of incorporating k elements of the perturbed vector v_i^k after the crossover operation into vector u_i^k follows a binomial distribution:

$$P(I = k) = \frac{n!}{k!.(n-k)!} Cr^k (1 - Cr)^{n-k}. \tag{15}$$

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

4.4. 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's motion model $x_{t+1}^i = f(x_t^i, u_t)$, the candidate pose, and the observed odometric data.

4.5. Additional mechanisms

The solution adopted in this work uses three main mechanisms to improve the robustness and efficiency of the basic DE algorithm. These mechanisms are:

(1) A *Threshold rejection factor* to avoid the premature elimination of solutions. This mechanism decreases the eagerness of the algorithm, allowing the elimination of a candidate solution from the set only when the offspring candidate is significantly better from a statistical point of view.
(2) A *Discarding operator* to accelerate the convergence of the algorithm by: (a) eliminating a very low percentage of the worst population individuals at each iteration of the algorithm, and (b) re-sampling the candidate individual in the vicinity of a better candidate selected randomly between the best elements of the population.
(3) An *Adaptive adjustment of the perturbation amplification factor F*. This mechanism tries to maintain a high amplification factor while the population has not converged to the promising areas (a wide scope search is required) and to limit the algorithm search scope when the population set is distributed in the most feasible areas.

4.6. Threshold determinations

4.6.1. L2-norm threshold

The fitness function value for a given candidate x_t^j (L2-norm, see Eq. (9)) is given by

$$f_0^j(x_t^j) = \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i}^j)^2}{2\sigma_e^2} + \frac{1}{2}(x_t^j - \hat{x}_t) P^{-1} (x_t^j - \hat{x}_t)^T \tag{16}$$

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

$$f_0^j(x_t^j) = \sum_{i=0}^{N_s} \frac{(z_{t,i} - \hat{z}_{t,i}^j)^2}{2\sigma_e^2} = \frac{1}{2} \sum_{i=0}^{N_s} \frac{v_{t,i}^2}{\sigma_e^2} \tag{17}$$

where $\nu_{t,i} = (z_{t,i} - \hat{z}^j_{t,i})$ 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 of $E[f_0]$ when the pose under evaluation is the true one. The term $\sum_{i=0}^{N_s} \nu^2_{t,i}/\sigma^2_e$, where $\nu_{t,i}/\sigma^2_e$ are the standard normal random variables $N(0, 1)$, is a chi-square distribution with N_s degrees of freedom. This distribution is well known and has mean N_s and variance $2N_s$. Then, the expected minimum fitness value will be

$$E[f_1] = \int_{-\infty}^{+\infty} f_0(\nu)p(\nu)\mathrm{d}\nu = N_s/2. \tag{18}$$

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 the 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 better fitness than the best pose existing 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.

4.6.2. L1-norm threshold

A similar approach can be done if the loss function selected is the L1-norm. In that case, the fitness function before the convergence point takes the following form

$$f^j_0(x^j_t) = \sum_{i=0}^{N_s} \frac{|z_{t,i} - \hat{z}^j_{t,i}|}{\sigma_e} = \sum_{i=0}^{N_s} \frac{|\nu_{t,i}|}{\sigma_e} \tag{19}$$

where $\nu_{t,i} = (z_{t,i} - \hat{z}^j_{t,i})$ represents the discrepancy between the observed and the predicted value of the sensor data. As in the L2-norm case, it is required to calculate the expected value $E[f_0]$ when the pose under evaluation is the true one and the noise is a normal distribution of zero mean and standard error deviation σ_e. The expected minimum fitness value can be easily calculated and will be

$$\begin{aligned} E\left[\sum_{i=0}^{N_s} |\nu_{t,i}|\right] &= \int_{-\infty}^{\infty} \sum_{i=0}^{N_s} |\nu_{t,i}| p(\nu_{t,i}) \mathrm{d}\nu_i \\ &= \sum_{i=0}^{N_s} \int_{-\infty}^{\infty} |\nu_{t,i}| \frac{1}{\sqrt{2\pi\sigma^2_e}} \mathrm{e}^{-\frac{1}{2}\frac{\nu^2_{t,i}}{\sigma^2_e}} \mathrm{d}\nu_i \\ &= 2\frac{1}{\sqrt{2\pi\sigma^2_e}} \sum_{i=0}^{N_s} \int_{0}^{\infty} |\nu_{t,i}| \mathrm{e}^{-\frac{1}{2}\frac{\nu^2_{t,i}}{\sigma^2_e}} \mathrm{d}\nu_i \\ &= 2\frac{1}{\sqrt{2\pi\sigma^2_e}} \sum_{i=0}^{N_s} \left[-\sigma^2_e \mathrm{e}^{-\frac{1}{2}\frac{\nu^2_{t,i}}{\sigma^2_e}}\right]_0^{\infty} \\ &= \sum_{i=0}^{N_s} \frac{2\sigma_e}{\sqrt{2\pi}} \end{aligned} \tag{20}$$

therefore

$$E[f_0] = \int_{-\infty}^{+\infty} f_0(\nu)p(\nu)\mathrm{d}\nu = \frac{2N_s}{\sqrt{2\pi}}. \tag{21}$$

4.7. Discarding operator

The inclusion of a threshold rejection mechanism in the selection operator of the DE algorithm increases the robustness of the algorithm, since the offspring poses accepted at the selection time have better statistical evidence of improving the parent pose. On the negative side, we can remark the decreasing effect on the convergence speed due to the high number of offspring poses rejected at the selection time, which originates that only a few number of offspring poses are included for the new population.

To accelerate the convergence speed a new operator has been included in the algorithm. The idea of this operator is to discard a low percentage δ of the worst elements of the population at each iteration. The discarded elements are substituted by a new pose chosen randomly between the β better elements of the population plus a random noise. To avoid a premature convergence the factors used in this work have been $\delta = 0.5\%$ and $\beta = 66\%$.

In the following section we will show the computational impact of the discarding operator and some guides to select the discarding parameters.

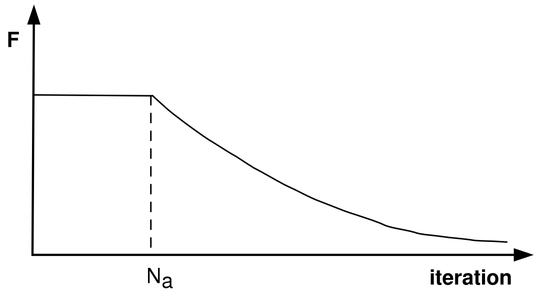


Fig. 2. Factor F curve as a function of the worst fitness value of the population set.

4.8. Adaptive amplification factor

The previous mechanisms improve the robustness greatly but do not exploit the local convergence. This effect is clearly evident in office buildings where many offices have the same dimensions, which originates multiple convergence areas. In the strongly multimodal situation two problems can appear. On the one hand, if we keep in the same place, it is necessary to maintain all the feasible areas until more discriminative probabilistic information is observed. To achieve this objective we limit the exploration capability of the algorithm to the neighborhood of the feasible areas detected (Fig. 2). On the other hand, when the robot gets out of the office to a corridor, if factor F is maintained high, the population tend to spread along the corridor areas (Fig. 3), and the limitation of the exploration decreases the spreading problem. The spreading is not a problem in global localization but creates serious disturbances in dynamical estimation problems.

To avoid both problems, the amplification factor is initialized at $F = 0.99$ in the first iteration of the observation cycle, and it remains in this value until the worst fitness value of the population reaches a pre-specified fitness function value. In our case we start the adaptation when the worst hypothesis in the set reaches four times the expected fitness value. After this level is reached, F is decreased a 0.5% at each iteration. This tends to focus the search area of the algorithm on the surroundings of the previous areas, avoiding an unnecessary dispersion of the population. The mechanism is started when the worst fitness of the population goes below two times the stopping criteria used. This mechanism improves the exploration of feasible detected areas at the final part of the algorithm iterations and is activated in the initial stages only when the algorithm has not converged to an only hypothesis.

4.9. Stopping condition

A classical problem in optimization methods is how to determine a stopping condition. This problem can be considered in

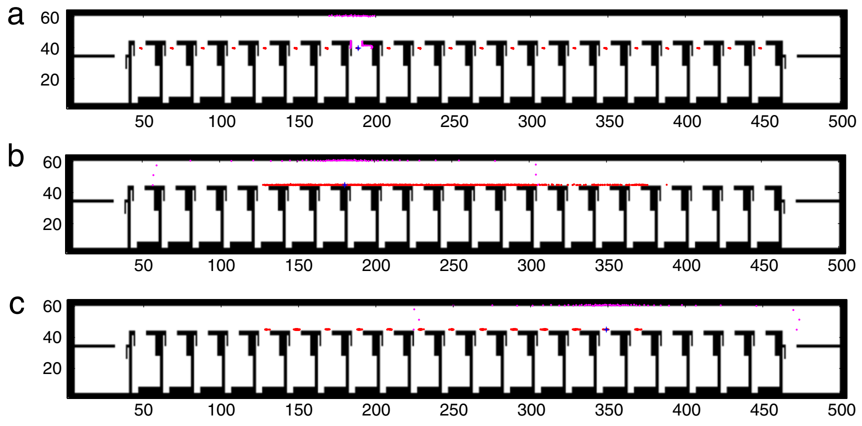


Fig. 3. Example of a spreading situation: (a) before spreading; (b) after spreading point maintaining F constant, and (c) after spreading point with F adaptation.

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, these conditions are not appropriate.

Assuming that the fitness function is 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 a $1-p$ probability to be inferior to the fitness function value at any perception cycle. The quantile values for some pre-specified probability values and degrees of freedom can be found in the statistics literature [26].

5. Control parameter election

Three main parameters control the behavior of the classical version of the DE algorithm: factor F (mutation control), factor CR (crossover control), and the population size. Different researchers have studied the influence of the control parameters in the performance of the DE algorithm, see for example [25,27,28]. The population size depends in general on a certain number of factors, among them: the number of dimensions of the problem, the size of the state space to be explored, and the kind of fitness function landscape, which is perhaps the most important one. The state estimation problems in dynamic systems add particular difficulties: the true state changes with the time, the fitness function also changes, and there is noise in the system and in the observations. All these factors affect the population required to maintain all feasible hypothesis until new information lets the algorithm converge toward the true one. In the problem under consideration in this work, the number of possible hypotheses is related to the map dimensions, the kind of environment, and the perceived information provided by the sensors. The less discriminative the information, the higher the number of possible hypotheses and consequently the population size required in the algorithm.

The adaptive adjustment of population size is currently under research in our group to avoid the typical trial and error adjustment process; but for the test environment used in the simulations (Fig. 10) a population of 300 elements has been adopted because it is enough to deal with non-degenerated perceptual situations. A degenerated perceptual situation happens when a robot located in a corner can only perceive a very limited part of the environment, which extremely increases the population set required since most corners are perceptually similar.

The other two factors have opposite effects: the selection mechanism tends to concentrate the population in the best areas reducing the diversity of the population and, on the other hand, the mutation increases the diversity of the population by exploring new areas. The premature convergence of the population is a classical problem of the evolutionary algorithms, and it is necessary to balance both effects to avoid this problem.

5.1. Election of factor F

The classical range of values for factor F is $(0, 1)$, although 1.0 is considered in practice an upper limit because most of the known functions that have been successfully optimized have not required values greater than one. However, this is not strictly a limit because it is possible to obtain solutions to different problems with $F > 1$, usually in a more time consuming and less reliable way than if $F < 1$. A particular situation happens when $F = 1$, since distinct vector combinations become indistinguishable, which reduces the number of possible vectors by a half and consequently reduces the exploration capability of the algorithm. Eq. (22) shows this situation where two combinations of the population vector lead to the same new trial solution; this effect can also be perceived in the curves of Fig. 4, where for F values close to the unity the average number of iterations up to convergence are very similar to the values obtained for $F = 0.5$.

$$v_i^k = x_{r_1}^k + (x_{r_2}^k - x_{r_3}^k)$$
$$v_i^k = x_{r_2}^k + (x_{r_1}^k - x_{r_3}^k). \qquad (22)$$

Fig. 4 shows the number of iterations up to convergence (obtained as the average best solution produced over 25 runs). The curves show the evolution of the number of iterations required for different F values under different observation noise to signal ratios (maintaining Cr constant and also the rest of factors). For increasing F values the number of iterations required to converge increases except in the vicinity of F due to the reason commented above. For values of $F > 1$ the increase in the number of iterations required to converge can be perceived. Though low F values provide low convergence times, they tend to be more easily trapped in local minima due to their small radius of exploration. Zaharie [28] has demonstrated the existence of a lower limit for F to operate efficiently, and has demonstrated that if F is too small, the population can converge even if the selection is turned off. For the dimension of the problem and the population we are considering here, this limit is very low, but undesirable effects start to appear for $F < 0.05$, where the convergence probability to the true pose decreases.

In our test, low F values are not convenient for the initial execution of the algorithm when the it needs to localize the feasible areas in the environment (the initial pose determination is similar to a noisy global optimization problem). However, in successive runs low F levels limit the mobility of the population,

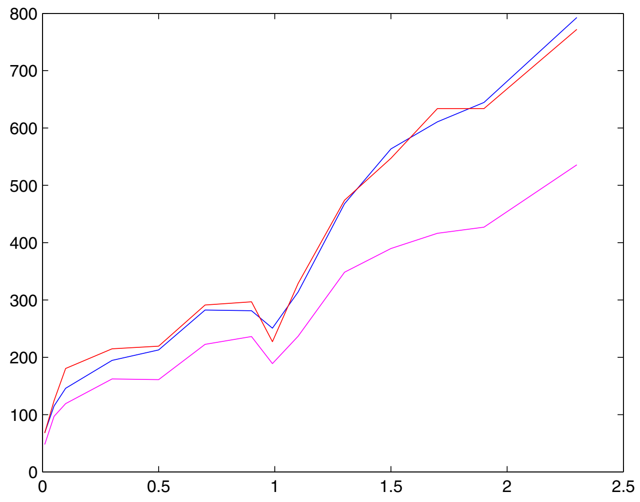


Fig. 4. Effect of the F values on the iterations number up to convergence ($CR = 0.5$) for a 1%, 5%, and 10% of noise standard deviation values over the measured signal values (magenta, red, and blue curves, respectively). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

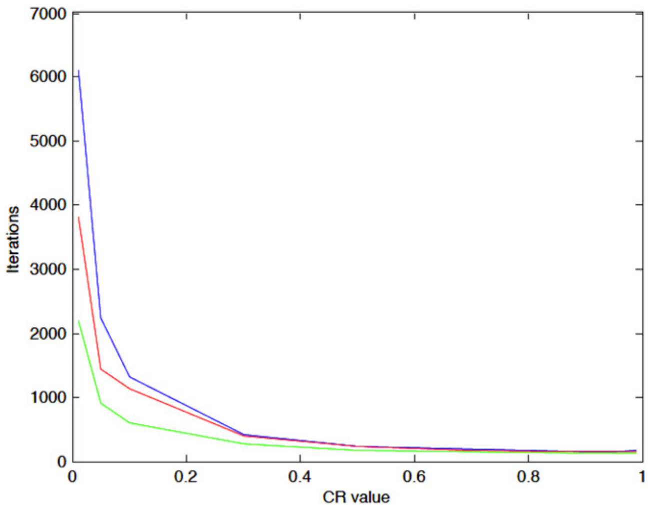


Fig. 5. Effect of CR values on the iterations number up to convergence ($F = 0.99$) for a 1%, 5%, and 10% of standard deviation values over the measured signal value (green, red, and blue curves, respectively). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

which is convenient for the long term efficiency and accuracy of the method (the tracking pose determination is basically a noisy local optimization problem in a dynamic system). The dispersion is inefficient in dynamic systems because they have history (except at the initial stage), and even if a dispersion is logical from a fitness point of view, it could not be logical from a dynamic point of view. This situation happens when a robot is located in an office and moves to the corridor, where many possible areas are perceptually equivalent, originating a dispersion in population that can lead to convergence problems.

According to that, an appropriate range for the F values can be $(0.5, 1.0)$ for the initial stages of the algorithm previous to the convergence to only one area, and lower levels but over the critical CR factor after the algorithm convergence. This results approximately coincide with those from other authors that suggest values in the range $0.4 < F < 0.95$, and some of them recommend $F = 0.9$ as a compromise between speed and probability of convergence in global non-separable optimization problems. In our simulation tests we have chosen $F = 0.99$. After the initial stages, once the feasible areas are reduced to one, the problem becomes a local optimization situation where the algorithm needs to track the feasible area while the robot is moving. For that situation a value $F = 0.05$ works efficiently, not exhibiting dispersion problems.

5.2. Election of factor CR

The crossover control parameter balances the importance given in the DE algorithm to the new trial solutions obtained from the perturbation mechanism (mutation) and the old solution. If $CR = 0$, the old population is maintained, and if $CR = 1$, the new trial solution is the mutated one. Values between both limits generate a trial set of solutions which combines the old solution with the mutated one, giving more probability to one or another depending on the value.

Fig. 5 shows the effect on the number of iterations required to reach the stopping condition at the initial perception cycle for the pose $(250, 100, 270)^T$ and with $F = 0.99$ (obtained as the average best solution produced over 25 runs). The effect of low CR values increases substantially the number of iterations required up to the stopping point due to the algorithm becomes less exploratory as CR decrease its value. The behavior of the algorithm is more evident when the noise level increases (blue curve).

Different works suggest the convenience of using low CR values when the objective function is separable, and high CR values for non-separable and multimodal objectives functions. In our case, values in the range $(0.5, 0.99)$ work efficiently, whilst lower CR values require higher number of iterations to reach the stopping criteria. $C_r = 0.5$ has been adopted in this work.

5.3. Election of the discarding factor

The discarding mechanism eliminates a fraction of the worst hypothesis in the population set and replaces it with a point in the neighborhood of one of the best solutions obtained up to that iteration in the current set. The new point is randomly chosen within the 66% of the best points in the population (according to its fitness) plus a small random noise. This mechanism tends to approach the worst solutions to the best areas and, as a consequence, it accelerates the convergence speed. This effect is shown in Fig. 6.

The results shown in Fig. 6 have been obtained using a population of 300 elements (also used along the rest of this work), $F = 0.99$, $C_r = 0.5$, and averaging the number of iterations required in 25 runs to reach the stopping condition of the algorithm for different rejection values. The number of rejected elements for a given population is obtained according to the expression $N_\delta = \text{round}(\delta * N_p)$. The curves obtained represent two highly different situations: pose $(x, y, \theta)^T = (250, 100, 270)^T$ (blue curve) is a distinguishable office located in the upper part of the corridor where the rooms are of different width (1 cell between two consecutive offices), and pose $(x, y, \theta)^T = (250, 30, 270)^T$ (red curve) is a non-distinguishable office located in the lower side of the corridor where the rooms are of the same width. In the second case there exist 21 rooms, which at initial observation provides equally probable hypotheses. Thus, the first run of the algorithm needs to be able to find the 21 equally probable areas to work in a robust way, while in the other initial pose the algorithm needs to find only one true area.

Fig. 6 clearly shows how the convergence time decreases substantially as the rejection factor increases. However, this mechanism modifies progressively the way of operation of the DE algorithm toward a less exploratory behavior, which is not convenient; and high discarding values originate the loss of

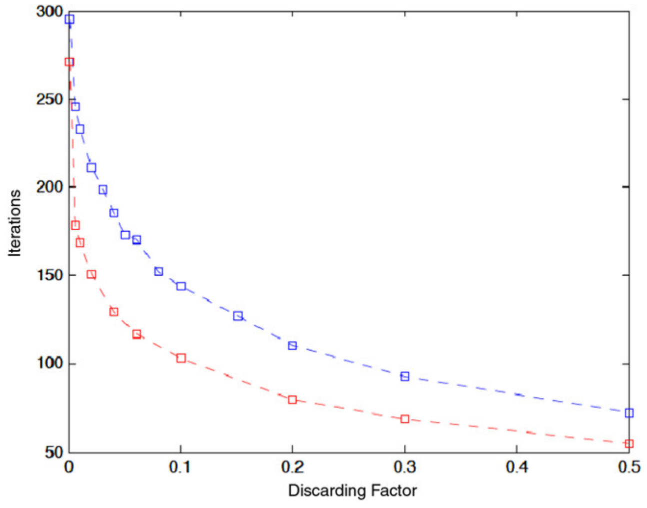


Fig. 6. Effect of the discarding factor on the iterations number up to convergence ($F = 0.99, Cr = 0.5$) for two different initial poses: $(250, 100, 270)^T$ and $(250, 30, 270)^T$ (blue and red curves, respectively). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

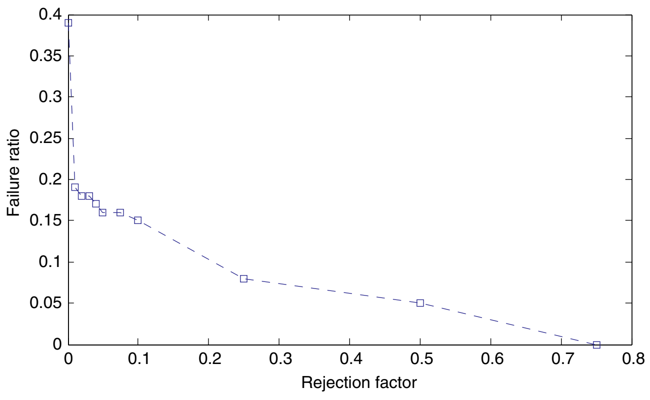


Fig. 7. Effect of the rejection factor on the hypothesis elimination up to convergence ($F = 0.99, Cr = 0.5$) for the initial pose $(250, 30, 270)^T$ (21 equally probable hypotheses).

feasible hypotheses in the multimodal case, which can decrease the robustness of the method. If a low discarding factor is adopted, the exploratory behavior of the DE method is not substantially altered and a shorter convergence time is reached. The curves show a fast decrease of the iterations up to convergence for discarding factors up to $0.05 * N_p$, and a slower decrease rate for higher discarding factors. In the multimodal case the loss of feasible hypotheses is perceived for discarding values over $0.1 * N_p$.

This figure also shows that the average iterations required for the algorithm up to the stopping point for the pose $(x, y, \theta)^T = (250, 100, 270)^T$ is 295.32 when no discarding is used; for $\alpha = 0.005$ the iteration number falls to 245.44; for $\alpha = 0.01$ – 233.24, and for $\alpha = 0.05$ – 172.92; that is, a 16.89%, 21.02%, and 41.44% of acceleration for $\alpha = (0.005, 0.01, 0.05)$, respectively. For $(x, y, \theta)^T = (250, 30, 270)^T$ the iterations number is 271.06 when no discarding is used, and 178.40, 168.60, and 129.60 for $\alpha = (0.005, 0.01, 0.04)$, respectively; this means a 34.18%, 37.79%, and 52.18% of improvement with respect to the non-discarding case. We have adopted a value of $\alpha = 0.05$ for the discarding parameter in order to achieve a reasonable acceleration without a substantial alteration of the method.

5.4. Election of the threshold rejection factor

The use of a threshold rejection factor (defined as a factor of the expected average value) tries to avoid a premature elimination of equally probable hypotheses. To observe the effect of different rejection factors, we have considered the robot located at pose $(250, 30, 270)^T$. Since there exist 21 similar offices and the orientation is opposite to the corridor door, all the offices are equally probable at the initial stage due to the fact that the perceptual information is exactly the same. Since the rejection and the stopping criteria affect the premature elimination of hypotheses, both effects have been checked separately. To test the effect of the rejection, the stopping condition used in the algorithm is set to stop when, after 100 iterations, no improvement has been achieved neither in the best estimate nor in the worst point of the population. The discarding has not been activated. In Fig. 7 the vertical axis shows the probability of losing one or more hypotheses at the stopping point obtained after 100 runs of the algorithm for different rejection factors (horizontal axis). It can be noticed that, for the stopping criteria defined, if no rejection is used, we have a probability of 0.38 of losing any of the feasible hypotheses. As the factor increases, this probability decreases. An interesting aspect is the strong effect obtained even for relatively low rejection factors: a $0.01 * f_{\text{expected}}$ decreases the probability to the half and for $0.75 * f_{\text{expected}}$ this probability goes to zero. The use of a rejection factor leads the algorithm to explore the space of possible solutions more exhaustively.

5.5. Election of the stopping criteria

The use of a statistical stopping criteria leads the algorithm to stop in a precise point (from a statistical point of view), which is considerably important in noisy dynamic systems where a lack of statistical uniformity at the stopping point can lead to prematurely eliminate feasible hypotheses or to a premature stopping. The premature stopping problem is for our dynamic state estimation problem of lower importance than the premature elimination of feasible areas, because the iterative execution of the algorithm at new observations let the method converge to the true pose progressively. However, this problem is undesirable because it increases the number of runs required to achieve the convergence to one pose. On the other hand, the premature elimination of feasible areas is a critical problem in our state estimation problem because once a feasible area is eliminated from the population, the probability of recovering it is very low and this can lead not to converge to the true pose.

Fig. 8 shows the probability of obtaining a number of hypotheses different from the feasible ones at the stopping point obtained after 100 runs of the algorithm. To obtain the curve the rejection and discarding are not activated, and different stopping factors have been checked (horizontal axis) for the pose $(250, 30, 270)^T$ previously commented. For relatively low factors the probability decreases very fast, reaching the zero probability for $4.0*f_{\text{expected}}$. If we increase the stopping factor, the number of iterations decreases and, after a point, the number of poses obtained increases since the algorithm does not have enough time to converge properly. Another effect is that both the number of iterations up to the stopping point and the accuracy of the solution obtained decrease as the factor increases.

From the analysis of the effect of the stopping and the rejection factors, it is possible to choose appropriate values for each one, but since both factors affect the probability of premature elimination, it is very interesting to observe the combined effect of both mechanisms. Fig. 9 shows the zero probability curve when using both mechanisms simultaneously. It can be noticed that low values are required for both parameters when used jointly. In our work the adopted values are $0.1*f_{\text{expected}}$ for the rejection factor and $1.0* f_{\text{expected}}$ for the stopping factor. This point is convenient because it maintains a good equilibrium between both mechanisms.

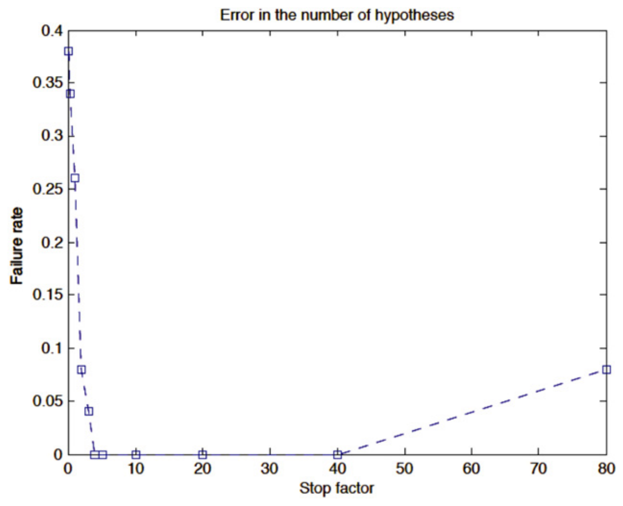


Fig. 8. Effect of the stopping factor on the hypothesis elimination up to convergence ($F = 0.99$, $Cr = 0.5$) for the initial pose $(250, 30, 270)^T$ (21 equally probable hypotheses).

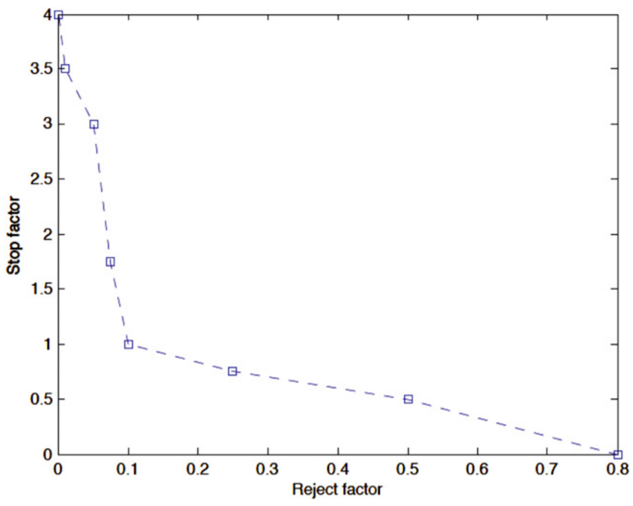


Fig. 9. Curve of zero probability of premature hypothesis elimination up to convergence ($F = 0.99$, $Cr = 0.5$) for the initial pose $(250, 30, 270)^T$ (21 equally probable hypotheses).

6. Experimental results

To test the algorithm characteristics, a simulated environment has been considered (Fig. 9). This environment is similar to many office indoor areas. All offices are located along the central corridor; the offices located in the upper part of the figure have the same length in y dimension and an x length progressively decreasing (in one cell unit) from offices located in the left side of the figure to those located in the right side. On the contrary, offices located on the lower part of the figure are of exactly the same dimensions and appearance. The offices located in the upper and lower corners of the environment have similar dimensions but doors are located in different sides.

This test environment will be used to check the accuracy and robustness properties of the L2-norm and L1-norm algorithms and the effect of the environment ambiguity on the algorithm convergence and response for different noise levels and different contamination levels of the noise error.

In Fig. 10 a localization sequence is shown. The robot is initially located at pose $(x, y, \theta) = (100, 100, 0)$ and turns over itself. This figure clearly shows how the method is able to determine the most probable areas (24 areas are maintained at the end of the first perception cycle) and how these multiple areas converge toward only one solution as the information perceived by the robot while it describes a turn is integrated. In this example, the population used is of 300 elements, which is enough for successfully representing most of the multi-hypothesis situations that can appear in this environment.

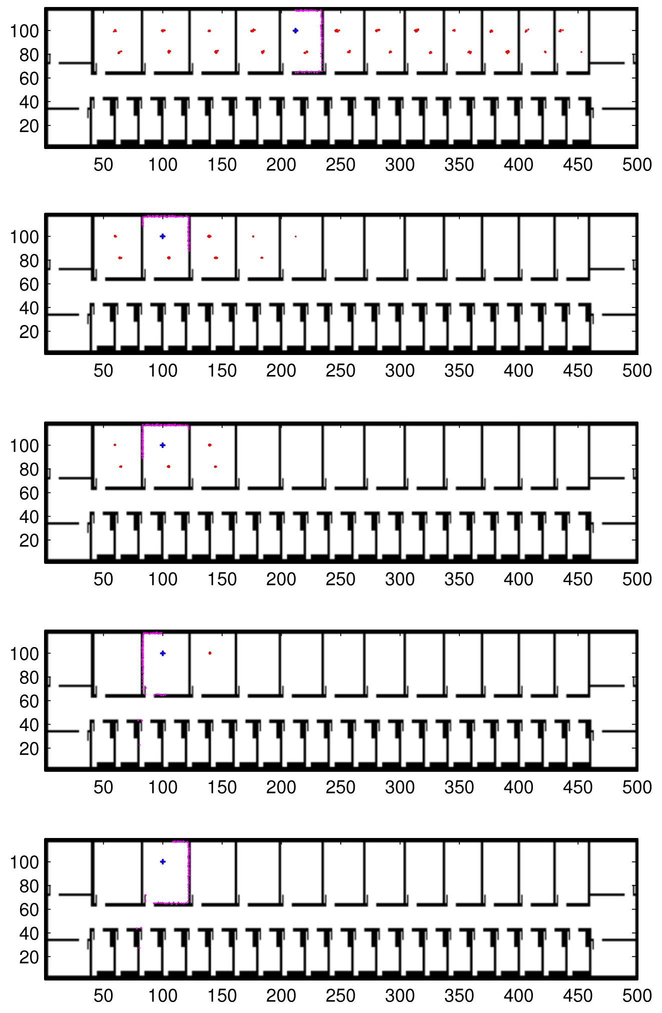


Fig. 10. Example of a localization sequence after perception cycles 1, 3, 5, 7 and 11, respectively.

6.1. Test 1

The first test tries to determine the capability of the algorithm to localize the robot when it is located in one of the distinguishable rooms of the upper side of the corridor and observes the distinctive characteristics of the room. The robot's pose is $(x, y, \theta)^T = (250, 100, 270)^T$ and the noise/signal ratio level has different values. We have considered a Gaussian observation error of zero mean and a σ value for each measurement proportional to a specified percentage of the measured distance. This kind of situation is typical when ultrasound sensors are used and is perhaps the worst case. When using laser range scanners the variance is almost constant, which constitutes a more favorable situation. In the test, the robot is located at a given pose and the algorithm is executed repeatedly maintaining that robot's pose but taking a new observation of the environment at each cycle. The results in Tables 1 and 2 show: the noise level (standard deviation expressed as a fraction of the signal measured); the average value of the absolute error in x, y, and θ, obtained for 50 runs of the algorithm (in cell units for x and y and degrees for θ (the cell size is

Table 1
L1-norm estimation error in cell units for different noise levels (cell size of 12 cm); true location: (250, 100, 270).

Noise	$\bar{\lvert e\rvert}_x$	σ_x	$\bar{\lvert e\rvert}_y$	σ_y	$\bar{\lvert e\rvert}_\theta$	σ_θ	*Av. cycle*	*Suc*
0.01	0.032	0.024	0.100	0.073	0.062	0.045	1	1.0
0.02	0.042	0.037	0.202	0.166	0.179	0.149	1.8	1.0
0.03	0.070	0.069	0.346	0.243	0.260	0.276	2.6	1.0
0.04	0.101	0.083	0.503	0.371	0.446	0.466	3.2	1.0
0.05	0.129	0.100	0.428	0.407	0.486	0.423	3.9	1.0
0.06	0.110	0.074	0.549	0.367	0.443	0.298	4.85	1.0
0.07	0.115	0.100	0.645	0.388	0.408	0.435	6.9	1.0
0.08	0.123	0.096	0.577	0.417	0.566	0.620	9.9	1.0
0.09	0.138	0.135	1.044	0.516	0.507	0.583	14.8	0.96
0.10	0.168	0.163	0.894	0.490	0.627	0.628	18	0.78

Table 2
L2-norm accuracy for different noise levels; true location: (250, 100, 270).

Noise	$\bar{\lvert e\rvert}_x$	σ_x	$\bar{\lvert e\rvert}_y$	σ_y	$\bar{\lvert e\rvert}_\theta$	σ_θ	*Av. cycle*	*Suc*
0.01	0.126	0.070	0.261	0.091	0.720	0.208	1	1.0
0.02	0.112	0.119	0.281	0.229	0.554	0.257	2	1.0
0.03	0.109	0.072	0.301	0.229	0.644	0.246	2.2	1.0
0.04	0.155	0.153	0.537	0.405	0.713	0.490	2.5	1.0
0.05	0.153	0.110	0.481	0.469	0.756	0.551	2.8	1.0
0.06	0.120	0.108	0.741	0.529	0.467	0.485	3.1	1.0
0.07	0.158	0.161	0.776	0.448	0.612	0.522	3.4	1.0
0.08	0.178	0.123	1.195	0.570	0.640	0.553	3.8	1.0
0.09	0.258	0.242	1.318	0.583	1.053	0.721	5.5	1.0
0.10	0.117	0.114	1.575	0.661	0.807	0.568	6.9	1.0

12 cm)); the standard deviation obtained for the absolute error in x, y, and θ; the average number of cycles required for the algorithm to converge to the true pose, and the success probability over the 50 runs for L1- and L2-norms.

According to Table 1, corresponding to the L-1 norm, it can be noticed that for a 1% of signal error variance, the mean of the absolute error in x is below 0.025 cell units (3 mm), below 0.100 cell units in y (12 mm), and below 0.07° in orientation. The variance is also very low. The accuracy keeps relatively constant between the 5% and the 8% of error signal variance. It does not decrease in that interval because the algorithm requires a growing number of perception cycles until it converges to one hypothesis, which lets the algorithm integrate more information and compensate the error increase. If the error is incremented, the accuracy starts to degrade slowly but the success ratio decreases. It is interesting to notice that, for an 8% of signal error variance, the L1-norm accuracy in x is around 1.5 cm, around 7 cm in y, and around 0.5° in orientation.

For a 1% of signal error variance the convergence to one hypothesis is achieved in one perception cycle, while for a 10% of signal error variance the convergence requires 18 perception cycles in average. In our tests, the success ratio of the L1-norm convergence to the true pose decreases very fast over the 10% of signal error variance.

Table 2 shows the same test for the L2-norm version of the algorithm. In this case, the algorithm maintains a relatively constant accuracy of 1.5 cm in x up to a 7% of variance level error. This does not happen in y dimension, where the average absolute error grows from around 3 cm up to 19 cm, while the attitude error shows a more constant accuracy between 0.5° and 1°, approximately. In terms of accuracy, the L2-norm shows less accurate results than the L1-norm case. On the other hand, the L2-norm requires less perception cycles to converge to the vicinity of the true solution and is able to provide a good estimate with higher level of noise than in the L1-norm case without degrading its success ratio. It is, from that point of view, more feasible than the L1-norm version of the algorithm.

6.2. Test 2

The second test tries to determine the capability of the algorithm to localize the robot when there exists a contaminated Gaussian noise. This situation happens when the robot tries to localize its pose in presence of mobile objects or unexpected obstacles. To test this situation the robot has been located in one of the distinguishable rooms of the upper side of the corridor and it observes the distinctive characteristics of the room, but the normal Gaussian noise has been contaminated with a uniform distribution located between the 25% and the 75% of the sensor measurement (around the middle of the real distance). This can be expressed by

$$p(x) = (1-\epsilon)N(x_m, \sigma) + \epsilon U(0.25x_m, 0.75x_m) \tag{23}$$

where ϵ is the contamination level, $N(x_m, \sigma)$ is the Gaussian observation noise probability distribution centered at the true measurement x_m, and $U(0.25x_m, 0.75x_m)$ is an uniform probability distribution between $[0.25x_m, 0.75x_m]$. The levels of contamination used for this test are 5% and 10%. The true robot's pose is $(x, y, \theta)^T = (250, 100, 270)^T$ and the Gaussian noise/signal ratio level has different values. We have considered a Gaussian observation error of zero mean and a σ value for each measurement proportional to a specified percentage of the measured distance. As in the previous test, Tables 3–5 show the performance of the algorithm for the L1- and L2-norms and different levels of error contamination.

For a 5% of noise contamination the important impact on the success ratio for the L2-norm can be noticed (Tables 3 and 4). In none of the Gaussian noise levels used for simulation a 100% of success ratio has been achieved, and for noise levels over 5% the success ratio decreases below a 50%. From a practical point of view, the L2-norm should not be used because, even in the best situation, the success ratio is below 90%. In the L1-norm case, the contamination affects the success ratio less seriously. For Gaussian noise levels up to 5% the success ratio of the L1-norm maintains the 100%, and for a 7% the success ratio is still over 70%.

Attending to the accuracy, for a 1% of variance in the Gaussian noise in presence of a 5% of contamination, the L2-norm has an

Table 3
L1-norm estimation error in cells for different noise levels and a 5% of contamination; true location (250, 100, 270).

Noise	$\bar{\lvert e\rvert}_x$	σ_x	$\bar{\lvert e\rvert}_y$	σ_y	$\bar{\lvert e\rvert}_\theta$	σ_{theta}	*Av. cycle*	*Suc*
0.01	0.023	0.019	0.150	0.113	0.070	0.065	1.7	1.0
0.02	0.059	0.043	0.210	0.143	0.206	0.200	2.4	1.0
0.03	0.103	0.088	0.361	0.277	0.374	0.402	2.85	1.0
0.04	0.117	0.118	0.512	0.401	0.430	0.387	3.7	1.0
0.05	0.091	0.088	0.766	0.544	0.497	0.547	4.85	1.0
0.06	0.097	0.105	0.814	0.500	0.640	0.474	6.24	0.84
0.07	0.139	0.146	1.079	0.435	0.516	0.464	11.3	0.70

Table 4
L2-norm estimation error in cell units for different noise levels and a 5% of contamination; true location: (250, 100, 270).

Noise	$\bar{\lvert e\rvert}_x$	σ_x	$\bar{\lvert e\rvert}_y$	σ_y	$\bar{\lvert e\rvert}_\theta$	σ_{theta}	*Av. cycle*	*Suc*
0.01	0.314	0.223	1.149	0.737	0.775	0.538	2	0.88
0.02	0.354	0.285	1.232	0.541	1.180	0.713	2.25	0.78
0.03	0.314	0.232	1.448	0.840	0.893	0.542	2.3	0.64
0.04	0.230	0.183	2.178	1.218	0.877	0.714	2.15	0.84
0.05	0.167	0.234	1.921	1.122	0.525	0.446	3	0.50
0.06	0.235	0.318	2.990	1.731	0.953	0.750	3.66	0.38
0.07	0.800	0.601	2.652	1.492	1.259	1.337	3.25	0.16

Table 5
L1-norm estimation error in cell units for different noise levels and for a 10% of contamination; true location: (250, 100, 270).

Noise	$\bar{\lvert e\rvert}_x$	σ_x	$\bar{\lvert e\rvert}_y$	σ_y	$\bar{\lvert e\rvert}_\theta$	σ_{theta}	*Av. cycle*	*Suc*
0.01	0.048	0.043	0.202	0.124	0.142	0.215	2.5	1.0
0.02	0.069	0.058	0.347	0.272	0.220	0.259	3.05	1.0
0.03	0.086	0.069	0.460	0.412	0.357	0.362	4.3	1.0
0.04	0.119	0.073	0.825	0.417	0.529	0.430	5.7	1.0
0.05	0.105	0.103	0.778	0.408	0.445	0.523	7.93	0.80
0.06	0.088	0.102	1.251	0.650	0.714	0.519	10.38	0.52

absolute error average of 0.314 cell units in x (3.76 cm), 1.149 cell units in y (13.78 cm), and 0.775° in attitude. These results deteriorate progressively with the increase of the noise variance. For a 1% of variance in the Gaussian noise the L1-norm error is of 0.023 cell units in x (0.27 cm), 0.150 cell units in y (1.8 cm), and 0.070° in attitude. These errors deteriorate as the variance of the Gaussian error noise increases, but for a 7% of variance the errors in x, y, and θ are 0.139 cell units, 1.079 cell units, and 0.516°, respectively. It is interesting to observe that, for this contamination level, the L1-norm accuracy results are only a little worse than the L1-norm results for the non-contaminated case. For the 7% of variance noise error in the non-contaminated case, the errors in x, y, and θ are 0.115 cell units, 0.645 cell units, and 0.408°, respectively, which are obviously better but still little accurate. It does not happen with the L2-norm results, where, even for the 1% of variance in the Gaussian error, the errors are substantially worse in x and y, and a little worse in θ. Besides, the accuracy of the L2-norm results deteriorates considerably as the variance in the Gaussian signal increases.

If we increase the contamination level up to a 10%, the L2-norm is not able to localize the robot (or in terms of success ratio, the probability of finding the true pose decreases below the 30% even for low variance levels in the Gaussian noise). In the L1-norm case (Table 5), the success ratio maintains a 100% up to a 4% of variance in the Gaussian noise and decreases over this value. The accuracy of the L1-norm is also affected, but for low levels of variance it is still highly accurate, e.g., for a 1% of variance the x, y, and θ average errors are of 0.048 cell units (0.57 cm), 0.202 cell units (2.42 cm), and 0.142°, respectively. The error levels increase with the variance.

6.3. *Test 3*

In Test 2 we have assumed a Gaussian noise contaminated by a uniform noise distributed around the middle range of the sensor measure. Test 3 tries to determine the capability of the algorithm to localize the robot in the presence of unexpected objects that contaminate the Gaussian sensor noise with an exponential distribution [13]. To test this situation the robot is located as in Test 2, but the normal Gaussian noise has been contaminated with an exponential distribution with $\lambda = 0.1$. This can be expressed by

$$p(x) = (1 - \epsilon)N(x_m, \sigma) + \epsilon\text{Exp}(\lambda) \tag{24}$$

where ϵ is the contamination level, $N(x_m, \sigma)$ is the Gaussian observation noise probability distribution centered at the true measurement x_m, and Exp(λ) is of the form

$$p_{\text{Exp}}(x) = \begin{cases} \lambda e^{-\lambda z_t^k} & \text{if } 0 \le z_t^k \le z_t^k * \\ 0 & \text{otherwise.} \end{cases} \tag{25}$$

This contamination distribution tends to be very aggressive due to the fact that the higher probability values tend to be concentrated near the robot, which originates big error values. The level of contamination used for this test is progressively increased. The true robot's pose is $(x, y, \theta)^T = (250, 100, 270)^T$. We have considered a Gaussian observation error of zero mean and a σ value for each measurement proportional to a specified percentage of the measured distance (the Gaussian noise/signal ratio level is assumed to be a 1%), an exponential contamination of $\lambda = 0.1$ and different contamination levels.

Similarly to the previous tests, Tables 6 and 7 show the results from the performance of the algorithm for the L1- and L2-norms and different levels of error contamination.

The exponential noise contamination has an important impact on the success ratio for the L2-norm, as can be observed in Table 6. In none of the exponential noise contamination levels used for simulation a 100% of success ratio has been achieved, and for contamination levels over a 5% the success ratio decreases below a 50%. From a practical point of view, the L2-norm should not be used because, even with a 1% of Gaussian noise in sensors, which

Table 6
L2-norm estimation error in cells for a Gaussian noise of 1% and different exponential contamination levels; true location (250, 100, 270).

Cont. level	$\|\bar{e}\|_x$	σ_x	$\|\bar{e}\|_y$	σ_y	$\|\bar{e}\|_\theta$	σ_{theta}	*Av. cycle*	*Suc*
0.01	0.2906	0.5638	1.1105	0.2840	0.8145	0.7760	1.0	0.96
0.02	0.2651	0.5122	0.8246	0.2001	0.7300	0.4901	1.0	0.88
0.03	0.5767	2.0096	1.5689	0.4840	1.8695	1.1077	1.0	0.72
0.04	0.4861	1.2343	0.7499	0.3971	1.6111	0.8825	1.0	0.52
0.05	0.5968	3.0364	1.8090	0.5339	1.9041	1.5062	1.0	0.44

Table 7
L1-norm estimation error in cell units for a Gaussian noise of 1% and different exponential contamination levels; true location: (250, 100, 270).

Cont. level	$\|\bar{e}\|_x$	σ_x	$\|\bar{e}\|_y$	σ_y	$\|\bar{e}\|_\theta$	σ_{theta}	*Av. cycle*	*Suc*
0.01	0.0385	0.1489	0.0660	0.0214	0.1027	0.0411	3.08	1.0
0.05	0.0350	0.1434	0.0695	0.0222	0.0984	0.0642	3.28	1.0
0.10	0.0546	0.1684	0.1698	0.0702	0.1079	0.2510	4.64	1.0
0.15	0.0339	0.2634	0.0788	0.0271	0.2150	0.0560	5.76	1.0
0.20	0.0486	0.1899	0.1029	0.0378	0.1661	0.1584	7.12	0.96
0.25	0.0529	0.3668	0.1850	0.0285	0.3193	0.1827	11.45	0.44

means a highly accurate sensor, a low exponential contamination of a 1% leads to a success ratio of only a 96%. In the L1-norm case, the contamination affects the success ratio less seriously. For exponential noise contamination levels up to 15% the success ratio of the L1-norm remains at a 100%, and for a 20% the success ratio is still over a 95%.

Regarding he accuracy, for a 1% of variance in the Gaussian noise and a presence of a 1% of exponential noise contamination the L2-norm has an absolute error average of 0.2906 cell units in x (3.487 cm), 0.5638 cell units in y (6.766 cm), and 1.1105° in attitude. These results deteriorate progressively with the increase of the contamination level. For the same conditions, the L1-norm has an error of 0.039 cell units in x (0.47 cm), 0.149 cell units in y (1.79 cm), and 0.07° in attitude. These errors do not deteriorate significantly as the contamination level increases. It is interesting to observe that, for these contamination levels, the L1-norm accuracy results are only a little worse than those for the non-contaminated case, but a higher number of perceptual cycles is required to reach the convergence to the true pose.

6.4. Computational cost

When analyzing the effect of the L1-norm and the L2-norm on the computational cost of the algorithm, three different situations have to be considered:

First perception 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 is around 8 s for the test example (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 (since the sensor perception estimation is done by ray tracing on the environment map, the estimation cost tends to grow with the size of the observed area). For the same population (300 points) and stopping criteria, this initial localization cycle can vary from 3 to 10 s. At the end of this first perception cycle the feasible localization areas are determined.

Perception cycles before convergence. This situation is faster than the initial localization cycle because the population set is distributed around the most favorable areas at the beginning of the perception cycle iterations and the stopping condition for the algorithm is reached considerably faster than in the initial case. For this test example the time required is around 200–400 ms.

Perception cycles after convergence. Once the algorithm detects that the whole initial population has converged to one hypothesis, 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 30–45 ms per iteration.

These times let the algorithm be used on-line except at the initial cycle. Due to the fact that the stopping condition definition has been done in statistical terms, there are no substantial computational differences at each perception cycle. The differences, however, are substantial, but they mainly affect to the number of perception cycles required for the population to reach the convergence point. In the favorable case that we are using to test and compare both norms, different situations can be noticed:

Gaussian noise. For low variance noise levels up to 3%, the number of perception cycles required to obtain the convergence is approximately similar, but after that point, the L1-norm requires more perception cycles. For a 10% of noise variance, the L1-norm requires 18 perception cycles and the L2-norm requires only 6.9 perception cycles in average. Both perception cycles requirements seem to grow exponentially, but the L1-norm grows faster. This fact makes the L1-norm be slower than the L2-norm when the noise variance increases.

Contaminated Gaussian noise. The contamination introduced in the perception noise slightly increases the number of perception cycles required to converge, but the general form of the computational cost growth is almost similar. It can be noticed that the L1-norm seems to be less affected than the L2-norm at low noise variance levels.

7. Conclusions

The Adaptive Evolutionary Localization Filter (A-ELF) presented in this paper is able to solve the global localization problem in an efficient way. The use of L1-norm or L2-norm loss functions in the algorithm affects the algorithm properties in the aspects commented next.

7.1. L2-norm properties

The use of the L2-norm has the following advantages:

The L2-norm is highly robust to increasing levels of variance in the non-contaminated Gaussian perception noise. The algorithm is able to localize the robot even with variance levels over a 10% of the measured distance.

The speed of convergence is excellent and faster than the L1-norm. That means that the algorithm requires less perception cycles to achieve the convergence to the true solution.

The accuracy of the algorithm is good. For a 1% of signal error variance in noise, the algorithm achieves an accuracy of 1.5 cm in x and 3 cm in y, while the attitude error accuracy is close to 0.5°, approximately.

The L2-norm is derivable and, for linear problems with Gaussian noise, leads us to closed estimator expressions (Kalman filter) with strong mathematical background referred to convergence and optimality proofs. In re-localization or tracking problems where the robot pose is known with some degree of uncertainty, these methods are fast and efficient because the localization problem is basically unimodal. However, this advantage is lost in the global initialization problem where the fitness function is multimodal. Up to day, it is still a very difficult task to model a multimodal probability distribution accurately, and consequently much more difficult if the multimodal probability distribution evolves and changes with the time. In these problems the mean and the variance are not good indicators.

Among the disadvantages of the L2-norm we have the following.

The L2-norm penalizes big errors strongly. This originates substantial problems when the perception noise has outliers or the perception is contaminated with some other probability distribution. In the tests, a 5% of contamination in the Gaussian noise is able to lead the algorithm to fail even with low variance levels in the Gaussian noise. The loss of properties in the L2-norm is very fast. In the tests, the success ratio for a 5% of contamination never reaches a 100% and decreases very fast as the Gaussian variance increases. For a 10% of contamination, the L2-norm success ratio is below 30% in all the tests. In case of exponential contamination the L2-norm degradation is faster because the unexpected measures are located close to the robot and the errors are bigger than in the uniform contamination.

The L2-norm accuracy decreases significantly in presence of contamination.

7.2. L1-norm properties

The use of the L1-norm has the following advantages:

The L1-norm is highly robust to increasing levels of variance in the non-contaminated Gaussian perception noise. The algorithm is able to localize the robot with variance levels up to 10% of the measured distance. It is less robust than the L2-norm to variance increases but still very robust.

The speed of convergence is good but not as fast as the L2-norm. That means that the algorithm requires more perception cycles to achieve the convergence to the true solution than the algorithm based on the L2-norm.

The accuracy of the L1-norm is considerably better than the L2 case. For a 1% of signal error variance, the mean of the absolute error is of 3 mm in x, 12 mm in y, and below 0.07° in orientation.

The L1-norm is robust to outliers or contaminated noise. The algorithm is able to cope with up to a 10% of contamination in the Gaussian noise without deteriorating the accuracy. For a 1% of signal error variance, the mean of the absolute error is of 0.57 cm in x, 2.4 cm in y, and 0.142° in orientation. The contamination tends to slightly decrease the accuracy and the range of variance where the algorithm is able to converge.

Among the disadvantages of the L1-norm we have the following.

The L1-norm requires more perception cycles to converge than the L2-norm.

The L1-norm tolerates lower levels of variance in the Gaussian noise than the L2-norm.

Apart from the differences obtained when using a loss norm or another, the algorithm presents some general characteristics:

(1) The algorithm can use different loss norms without problems, since the method is based on a multi-point stochastic search. The use of a stochastic search method lets the algorithm operate with non-derivable loss functions that cannot be solved analytically. Besides, it can work with arbitrary nonlinear system dynamics, sensor characteristics, and non-Gaussian noise.

(2) Since the algorithm searches for the set of solutions according to the current loss function and it does not try to approximate posterior density distributions, it does not require any assumptions on the shape of the posterior density as parametric approaches do. This avoid the slow convergence to feasible areas obtained with Monte Carlo methods.

(3) The size of the minimum solution set required to guarantee the convergence to the true solution is very small. For the test shown previously, a population of 300 elements has been adopted because it is the one required to localize an arbitrary informative pose, but for the pose tested 30 elements are enough. The minimum set of points required to achieve a successful global localization for an arbitrary point depends on the environment information observed. If this information is scarce (e.g., when the robot is cornered and can only observed the corner), the number of elements grows. This is because the algorithm requires a minimum number of points (10–15, approximately) to efficiently manage each feasible hypothesis, and the number of feasible hypotheses is proportionally inverse to the environment information perceived. The number of points can be decreased when the number of hypotheses decreases.

(4) The algorithm is easy to implement and the computational cost makes it able to operate on-line even in relatively big areas. In spite of its computational efficiency, it is not a closed analytical solution to an optimization problem and, consequently, it is slower than closed solutions. But closed solutions cannot deal with multi-hypotheses (multi-solutions) problems, and those methods require additional algorithms to cope with this situation. This fact leads them to loose their initial speed and theoretical advantages.

(5) In the method some mechanism have been introduced. The discarding is applicable to accelerate most of optimization methods while the thresholding is mainly oriented to be used in state estimation in dynamical problems subject to noise. The last mechanism, the F decreasing and the posterior use of low F values together with a local search is also oriented avoid losing hypotheses in multimodal situations and to decrease the spreading problem in state estimation in dynamical problems subject to noise.

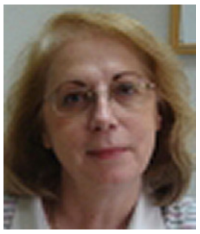
References

[1] H.G. Wilson, Least squares versus minimum absolute deviations estimation in linear models, Decision Sciences 9 (2) (1978) 322–335.
[2] S.C. Narula, P.H.N. Saldiva, C.D.S. Andre, S.N. Elian, A.F. Ferreira, V. Capelozzi, The minimum sum of absolute errors regression: a robust alternative to the least squares regression, Statistics in Medicine 18 (11) (1999) 1401–1417.
[3] S.C. Narula, J.F. Wellington, The minimum sum of absolute errors regression: a state of the art survey, International Statistical Review/Revue Internationale de Statistique 50 (3) (1982) 317–326.
[4] J.R. Rice, J.S. White, Norms for smoothing and estimation, SIAM Review 6 (1964) 243–255.
[5] R. Blattberg, T. Sargent, Regression with non-Gaussian disturbances: some sampling results, Econometrica 39 (1971) 501–510.
[6] H. Ekblom, $L(p)$ methods for robust regression, BIT 14 (1974) 22–32.
[7] T. Dielman, A comparison of forecasts from least absolute value and least squares regression, Journal of Forecasting 5 (3) (1986) 189–195.
[8] D. Fox, W. Burgard, S. Thrun, Markov localization for mobile robots in dynamic environments, Journal of Artificial Intelligence Research 11 (1999) 391–427.
[9] W. Burgard, D. Fox, D. Henning, T. Schmidt, Estimating the absolute position of a mobile robot using position probability grids, in: Proc. of the National Conference on Artificial Intelligence AAAI-96, Portland, Oregon, USA, 1996, pp. 896–901.
[10] J. Reuter, Mobile robot self-localization using PDAB, in: Proc. IEEE International Conference on Robotics and Automation, ICRA-2000, San Francisco, CA, 2000.
[11] P. Jensfelt, O. Wijk, D.J. Austin, M. Andersson, Experiments on augmenting condensation for mobile robot localization, in: Proc. of the Int. Conference on Robotics and Automation ICRA-00, San Francisco, CA, USA, 2000, pp. 2518–2524.
[12] F. Dellaert, D. Fox, W. Burgard, S. Thrun, Monte Carlo localization for mobile robots, in: Proceedings of the 1999 International Conference on Robotics and Automation, 1999, pp. 1322–1328.

[13] S. Thrun, D. Fox, W. Burgard, F. Dellaert, Robust Monte Carlo localization for mobile robots, Artificial Intelligence 128 (2001) 99–141.
[14] L. Moreno, S. Garrido, M.L. Muñoz, Evolutionary filter for robust mobile robot localization, Robotics and Autonomous Systems 54 (2006) 590–600.
[15] F. Martín, L. Moreno, S. Garrido, D. Blanco, Localization in 3D environments using differential evolution, in: 2009 IEEE International Symposium on Intelligent Signal Processing, WISP'2009, Budapest, Hungary, August 2009.
[16] M. Lisowski, Differential evolution approach to the localization problem for mobile robots, Master Thesis, Technical University of Denmark, September 2009.
[17] A.R. Vahdat, N.N. Ashrafoddin, S.S. Ghidary, Mobile Robot global localization using differential evolution and particle swarm optimization, in: Proc. IEEE Congress on Evolutionary Computation, CEC-2007, 2007, pp. 1527–1534.
[18] I.J. Cox, J.J. Leonard, Modeling a dynamic environment using a Bayesian multi hypothesis approach, Artificial Intelligence 66 (1994) 311–344.
[19] P. Jensfelt, S. Kristensen, Active global localisation for a mobile robot using multiple hypothesis tracking, in: Proc. IJCAI Workshop on Reasoning with Uncertainty in Robot Navigation, Stockholm, Sweden, 1999, pp. 13–22.
[20] D.J. Austin, P. Jensfelt, Using multiple Gaussian hypotheses to represent probability distributions for mobile robot localization, in: Proc. of the Int. Conference on Robotics and Automation ICRA-00, San Francisco, CA, USA, 2000, pp. 1036–1041.
[21] S.I. Roumeliotis, G.A. Bekey, 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, 2000, pp. 2985–2992.
[22] K. Arras, J.A. Castellanos, R. Siegwart, 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, Washington, DC, USA, 2002, pp. 1371–1377.
[23] R. Storn, K. Price, Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, March 1995.
[24] R. Storn, K. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (1997) 341–359.
[25] K. Price, R. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, 2005.
[26] M.R. Spiegel, J.J. Schiller, R.A. Srinivasan, Probability and Statistics, McGraw-Hill, 2000.
[27] J. Liu, J. Lampinen, On setting the control parameters of the differential evolution method, in: Proceedings of the 8th Int. Conference on Soft Computing, MENDEL, Brno, Czech Republic, 2002, pp. 11–18.
[28] D. Zaharie, Critical values for the control parameters of differential evolution algorithms, in: Proceedings of the 8th Int. Conference on Soft Computing, MENDEL, Brno, Czech Republic, 2002, pp. 62–67.

L. Moreno received the Degree in Automation and Electronics Engineering in 1984 and the Ph.D. degree in 1988 from the Universidad Politécnica de Madrid, Madrid, Spain. From 1988 to 1994, he was an associate professor at the Universidad Politécnica de Madrid. In 1994, he joined the Department of Systems Engineering and Automation, Universidad Carlos III de Madrid, Madrid, Spain, where he has been involved in several mobile robotics projects. His research interests are in the areas of mobile robotics, mobile manipulators, environment modeling, path planning and mobile robot global localization problems.

D. Blanco received the B.S. degree in Physics from the University Complutense of Madrid, Spain in 1992. She received the Ph.D. degree in Mecatronics from University Carlos III of Madrid (UC3M) in 2002. From 1996 to 1999, she was a fellowship student at the Department of Systems Engineering and Automation of UC3M. Since 2009, she is associated professor in the same Department. Member of Mobile Manipulator Group at UC3M. Her current research includes sensor based path planning, localization and control for mobile manipulators.


M.L. Muñoz was born in Madrid, Spain. She graduated in physics science from the Complutense University of Madrid and received the Ph.D. degree in 1988. During her predoctoral period she collaborated as a researcher at the Scientific Research Council (CSIC), Spain. Since 1990, she is an Associate Professor of Computer Technology and Architecture at the Polytechnic University of Madrid, Spain. She has been researching in a broad range of topics that covering signal processing, neural network algorithms, image processing and computational vision. Nowadays, her main research interests include evolutionary algorithm applications to mobile robotics.

S. Garrido received the Degree in Mathematics in 1979 from the Complutense University of Madrid, the Degree in Physics in 1955 and the Ph.D. degree in 2000 from the Universidad Carlos III de Madrid, Spain. In 1997, he joined the Department of Systems Engineering and Automation, Universidad Carlos III de Madrid, Madrid, Spain, where he has been involved in several mobile robotics projects. His research interests are in the areas of mobile robotics, mobile manipulators, environment modeling, path planning and mobile robot global localization problems.