2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

A Spatially Structured Genetic Algorithm over Complex Networks for Mobile Robot Localisation

Andrea Gasparri, Stefano Panzieri, Federica Pascucci, and Giovanni Ulivi

***Abstract*— One of the most important problems in Mobile Robotics is to realize the complete robot's autonomy. In order to achieve this goal several tasks have to be accomplished. Among them, the robot's ability to localise itself turns out to be critical. The research community has provided, through the years, different methodologies to face the localisation problem, such as the Kalman Filter or the Monte Carlo Integrations methods. In this paper a different approach relying on a specialisation of the genetic algorithms is proposed. The novelty of this approach is to take advantage of the complex networks theory for the spatial deployment of the population to more quickly find out the optimal solutions. In fact, modelling the search space with complex networks and exploiting their typical connectivity properties, results in a more effective exploration of such space.**

I. INTRODUCTION

Reliable pose information is fundamental to achieve the complete autonomy of a mobile robot moving in a known environment. The localisation problem aims to estimate the robot's pose using data coming from sensors: noisy data and unpredictable interactions with the environment itself makes the problem considerably difficult.

Localisation is usually divided into three different research problems: position tracking, global localisation, and kidnap. The majority of the works in literature is based on the probabilistic framework that recursively update a probability distribution, called *Belief*, over all space points in the environment. In this framework the position tracking, i.e., estimating the robot's pose with a prior knowledge about the initial robot's location, can be solved by the Kalman Filter approach [1].

Global Localisation is the problem of estimating the robot's pose without benefit of *a priori* knowledge of initial robot's location. This lack of knowledge makes the problem even more difficult as environmental ambiguities have to be carefully considered in order to successfully determine the robot's pose from scratch. The Kalman Filter approach, unimodal in nature, cannot be successfully applied in this case due to the necessity of tracking multi-hypotheses. Relaxing the Gaussian assumption, other probabilistic global techniques have been proposed. For instance, in [2] a grid-based discretisation of the state space has been used to approximate a more complex set of distributions. However, it suffers from excessive computational overhead [3]. A more promising approach is based on sequential Monte Carlo integration methods [4] that use a set of random weighted samples (particles) to approximate the probability distribution. The advantage is clearly in the possibility of representing a large number of probability distributions, but a very high number of particles have to be used to get a better approximation.

A last problem arises when considering the kidnapping of a well localised robot. New data may indicate a completely different position, and algorithms should manage several hypotheses in order to properly recover the robot's pose. Unfortunately, Monte Carlo integration methods suffer from *degeneracy problem*, i.e., the problem of having most of the particles with a negligible weight after few iterations [5]. Therefore the possibility to maintain multi-hypotheses is no longer guaranteed.

A possible solution to all three problems has been proposed in [6]. Here, a spatial clustering procedure along with a genetic algorithm has been proposed. In particular, the former is used to smartly partition the population set, while the latter is applied within each subset in order to more quickly find out local minima.

In this paper, moving from the experiences of [6], a new approach, based on spatially structured genetic algorithms (SSGA) [7], will be presented. A genetic algorithm for mobile robot localisation can be already found in [8]. Here, an integration between an EKF with a genetic optimisation filter is proposed. At each time-step, the EKF is used to predict the new state and to determine the new search radius where the genetic optimisation filter is applied to improve the estimate. Conversely, the approach proposed by the authors relies on the fact that the introduction of spatial structures in evolutionary algorithms helps to create evolutionary niches that can be regarded as regions which preserve solutions that could be useful in the future [9]. In particular, the proposed spatial structure is obtained by means of some complex networks: the Watts-Strogats [10] model and the scale-free Barabási-Albert model [11]. These models show some peculiarities from a topological point of view, for example, the small world property.

In section II a review of the two models is reported along with an SSGA introduction. The influence of a network structure, which is also analysed in [12], will be discussed in detail in section III, where a new competition rule between individuals will be introduced together with a kidnap recognising procedure. In section IV some experimental results, obtained with a mobile robot moving in a real environment, will be shown to demonstrate the effectiveness of our approach.

All the authors are with the Dip. di Informatica e Automazione, Università "Roma Tre", Via della Vasca Navale, 79, 00146 Roma, Italy (www.dia.uniroma3.it)

1-4244-0602-1/07/$20.00 ©2007 IEEE.

II. THEORETICAL BACKGROUND

A. Complex Networks

Complex Networks are graphs of nodes or vertices connected by links or edges, currently used to describe many natural or artificial systems: the brain, for instance, can be modeled as a network of neurons, and the Internet as a complex network of routers and computers linked by several physical means.

From the beginning, complex networks have been investigated by the graph theory community, who proposed several models, such as regular and random graphs; since then several other communities have been interested in this topic. Today, a main research issue is to figure out the relationship between structural and dynamic properties of the networks.

Regular graphs, introduced to describe systems made of a limited number of nodes, were revealed to the research community to be inadequate with the appearance of large-scale networks. This has lead the community to focus their attention on random graphs.

According to [13], once the probability p of having a connection among pairs of nodes is fixed, a random graph with N nodes and about $pN(N-1)/2$ links, can be obtained randomly selecting a pair of nodes and linking them with such probability p. This model has been extensively used since particular properties of complex networks, such as the small-world property or the scale-free one, have been discovered.

To better understand such properties, some basic concepts about complex networks have to be introduced: the *average path length*, the *cluster coefficient* and finally the *degree distribution*.

The *average path length* L of the network is the mean distance between two nodes, averaged over all pairs of nodes, where the distance between two nodes is defined as the number of the edge along the shortest path connecting them.

The *cluster coefficient* C of the network is the average of C_i over all nodes i, where the coefficient C_i of node i is the average fraction of pairs of neighbours of the node i that are also neighbours of each other.

The *degree distribution* of the network is the distribution function $P(k)$ describing the probability that a randomly selected node has exactly degree k, where the degree k is the number of links a node owns.

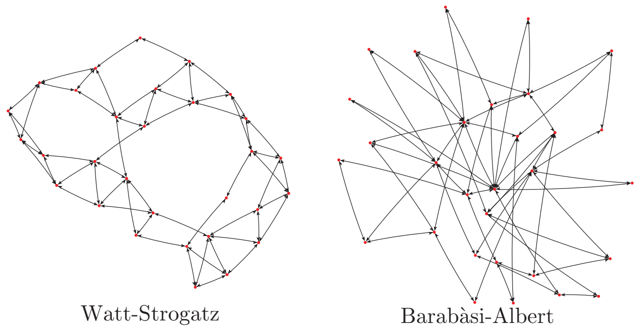

Fig. 1. Watt-Strogatz and Barabási-Albert models with 30 nodes

From a formal point of view, regarding these basic properties, several complex network models can be correctly defined. Regular graphs, for instance, are characterised by a high cluster coefficient, approximately $C \cong 3/4$ and a large average path tending to infinity as $N \to \infty$. Random graphs have a low cluster coefficient, approximately equal to the probability p defined above, and a short average path $L_{aver} \cong lnN/(pN)$.

In [10] the *Small-World* model is proposed to better describe real systems. It shows properties of both the regular and random graphs, such as a high cluster coefficient and a short average path, underlining the fact that, in reality, the circle of acquaintances of people is not only restricted to their neighbours.

In [11] the *Scale-Free* model is presented. This model, relying on the power-law degree distribution, overcomes the limitations of the previous ones through a hierarchical description of nodes. As a consequence, in a scale-free network preferential attachments are possible. This model, for instance, turns out to be very useful to describe airline routing maps.

Two examples of the above described networks are reported in Fig. 1; for a complete overview of complex networks the [14] is suggested.

B. Spatially Structured Genetic Algorithms

Genetic algorithms are a class of research techniques, inspired by Darwin's *Theory of Evolution*, applied in several research fields to solve optimisation problems. These algorithms use a population of encoded strings (*chromosomes*) as candidate solutions to explore the search space. The candidate's evaluation is performed by means of an objective function (*fitness function*) and improvements at each iteration (*epoch*) result from the application of probabilistic transition operators (*crossover* and *mutation*) acting onto chromosomes. A simple genetic algorithm (SGA) usually provides three steps: initialisation, selection and reproduction [15]. Initialisation generates a population randomly picking up elements over the whole search space, selection draws an intermediate population relying on a fitness-based approach and reproduction causes the population to evolve combining elements from the intermedia population. The pseudocode in Algorithm 1 shows a possible implementation schema for an SGA, where the roulette wheel selection along with crossover and mutation are adopted.

Usually, crossover is performed with probability p_{cross}, while *mutation* modifies chromosomes with probability p_{mutat}. This means that some individuals, likely with high fitness, will be exactly copied in the new population.

A spatially structured genetic algorithm (SSGA) is a specialisation of an SGA, where the population is spatially distributed with respect to some discrete topology. If the topology is a network these methods are also known as *graph based genetic algorithms* [7]. Following the latter approach, a population P can be defined through a set $V = \{p_1, \dots, p_n\}$ of vertices and an incidence matrix $M = \{(i, j) = 1 :$

Algorithm 1: A Simple Genetic Algorithm Schema

```
Data: Fitness function $f(\cdot)$
Result: $p^*_k$
/* Initialisation                                      */
Set $k = 0$; Create $P_k = \{p_{1,k}, \ldots, p_{n,k}\}$
while $StopCondition(\tilde{p}_{j,k})$ do
    /* Roulette Wheel Selection                        */
    for i=1 to $n$ do
        $x = random(0, 1)$;
        $j = 1$;
        while $j < n$ & $x < \frac{\sum_{l=1}^{j} f(p_{l,k})}{\sum_{l=1}^{n} f(p_{l,k})}$ do
            $j = j + 1$;
        end
        $\tilde{p}_i = p_{j,k}$;
    end
    /* Crossover Reproduction                          */
    for i=1 to n-1 do
        $p_{i,k+1} = Crossover(\tilde{p}_i, \tilde{p}_{i+1})$
    end
    /* Mutation Reproduction                           */
    for i=1 to $n$ do
        $p_{i,k+1} = Mutation(\tilde{p}_i)$
    end
    $p^*_{k+1} = max_{f(\cdot)}\{\{p_{1,k+1}, \ldots, p_{n,k+1}\}\}$;
    $k = k + 1$;
end
```

$\exists\ link\ between\ i\ and\ j\}$ as:

$$P := \{V, M\}. \tag{1}$$

According to this structure, selection picks up pairs of vertices which show a relationship into the incidence matrix M, and reproduction generates new elements preserving the network topology.

Therefore, differences between SGA and SSGA are mainly related to the selection approach: the former performs this step by means of a fitness-based approach, such as the roulette wheel, whereas the latter exploits the network topology of the population.

III. The proposed spatially structured genetic algorithm over complex networks

The proposed SSGA takes advantage of the complex networks theory, deploying a population over such topologies, to more quickly discover the optimal solution. In particular, in this paper the connectivity properties of different complex network models have been exploited for a more effective exploration of the search space.

The algorithm follows the classical SGA schema previously shown with a specialisation for each step. Initialisation creates a population over a complex network; in particular, two different kinds of topologies have been exploited, the *small-world* and the *scale-free* models. Selection picks up all pairs of nodes that, based on the incidence matrix M, are linked, and reproduction compares each pair of elements to decide which probabilistic transition operator needs to be used. For this comparison, the average fitness value over the whole population is used as a threshold. As a consequence two possible states for each element can be defined, *high* or *low*.

In this context, the full state of the robot (x, y, θ) is used as chromosoma, while the pattern function $Pat(z_k, x_k)$ is used as fitness function. In detail, the pattern function gives a measure of the similarity between two vectors, as follows:

$$Pat(z_k, \hat{z}_k) = \frac{1}{L}\sum_{i=1}^{L}\frac{1}{\sqrt{2\pi\sigma}}e^{\frac{-(z^i_k - \hat{z}^i_k)^2}{2\sigma}} \tag{2}$$

where, z_k represents the sensor data, $\hat{z}_k$ is the expected one for the considered hypothesis and finally σ is a measure of confidence.

Now, three cases can arise when two elements are compared: high-high, high-low, and low-low. A well-defined action (local mating rule) is then here proposed for each one:

- high-high: the lowest element is replaced with the result of the crossover;
- high-low: the lowest element is replaced with a mutated version of the highest one;
- low-low: both of the elements are mutated.

Regarding the probabilistic transition operators: *crossover* picks up two elements and performs a convex combination of them with probability p_{cross}, while *mutation* picks up an element and modifies its chromosoma, inversely proportional to its fitness, with probability p_{mutat}.

Two remarks are now in order:

- After the crossover operator, two mutations with lower probabilities are tried to allow a better exploration of the space around good solutions.
- Each new element created by the mutation is compared with the original one in order to be accepted. In such a way the research is partially inhibited, and the exploration of the space does not destroy solutions that are good enough.

Although the algorithm is able to solve the global localisation problem with a proper random initialisation, an additional strategy, able to *sense* when a kidnap occurs, has to be provided in order to spread the population over the search space again and then re-localise the robot.

In this paper, the fitness function $f(\cdot)$ and the edge function $e(\cdot)$, i.e., the fraction of potential mating couples (couples with different genotypes) over the network (the number of links) [7], are used to trigger the spreading action. The edge function gives an evaluation of the dispersion of the population. For a well-localised robot, a high percentage variation of the fitness along with a considerable dispersion of the population (a high value of the edge function), are reliable symptoms of a kidnap.

The pseudocode in algorithm 2 shows a possible implementation schema for a generic step k of the proposed SSGA.

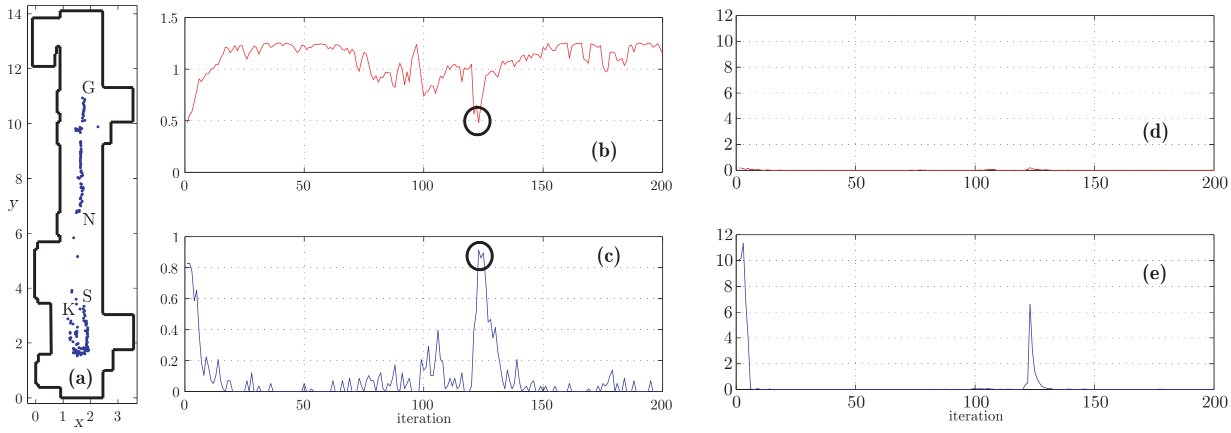

Fig. 2. (a) Map of the first environment and path of the robot (S:start, K:kidnap point; N:new start; E: end of path) - (b) Fitness function - (c) Edge function - (d) Dispersion along x axis - (e) Dispersion along y axis

IV. Simulation and Experimental Results

The proposed SSGA algorithm was first tested in a simulated environment to properly set the network parameters. Afterward, experiments with real robot data have been performed to prove its effectiveness.

A. Robot and Sensor Modelling

The robot pose, denoted by the state variable x, can be entirely described on a plane by its position (p_x, p_y) and its orientation ϕ. Here, the unicycle model has been adopted as kinematic model for the robot. Specifically, $u_{k-1} = (\delta s_k, \delta\phi_k)$ is the system input, where δs_k is the vehicle displacement, and $\delta\theta_k$ the rotation during the sample time interval δt_k, both measured by proprioceptive sensors. As a consequence, the system model equation is:

$$\begin{aligned} x_k &= f(x_{k-1}, u_{k-1}, n_{k-1}) \\ &= x_{k-1} + \begin{bmatrix} \cos\tilde{\phi}_{k-1} & 0 \\ \sin\tilde{\phi}_{k-1} & 0 \\ 0 & 1 \end{bmatrix} u_{k-1} + n_{k-1} \end{aligned} \qquad (3)$$

where $\tilde{\phi}_{k-1} = \phi_{k-1} + \delta\phi_{k-1}/2$ is the average robot orientation during the sample time interval δt_k, and n_{k-1} is a white zero mean noise.

The robot, that is moving in an environment completely described by a list $\mathcal{M}$ of pairs of points, has been equipped with L laser range finders arrayed on 360°. The resulting observation model for each laser beam is:

$$z_{j,k} = h(x_k, \mathcal{M}) = \frac{|a_r l_j^x + b_r l_j^y + c_r|}{|a_r \cos\theta_j + b_r \sin\theta_j|} \qquad (4)$$

where (a_r, b_r, c_r) are the coefficients of the r-th segment and (l_j^x, l_j^y, θ_j) is the configuration of the laser beam detecting the segment considered.

B. Simulations

Simulations have been done using a framework, developed by the authors on Matlab, with the ability to provide several robot kinematics and an emulation for several sensors, among them laser and sonar rangefinders. A first analysis has been performed with the aim of calibrating network parameters. In particular, a desired behaviour is the one that allows the persistence of several evolutionary niches, corresponding to several hypotheses, without preventing the convergence to the best one.

Such analysis has pointed out, for the Watt-Strogatz model, that a network degree equal to 3 and a rewiring probability of 0.1 are a satisfactory set of parameters; a similar analysis, reported in [16], shows that for these values, such a network shows a high clustering coefficient and a small average path length. This condition increases the selection pressure (the speed of convergence is high but maximum) along with the persistence of niches.

For the scale-free Barabási-Albert model the behaviour is slightly different due to the presence of *hubs*, i.e., highly connected nodes. Scale-free networks have a shorter takeover time (the time it takes for the single best individual to conquer the whole population), that could lead the SSGA to converge too rapidly towards an incorrect solution in a noisy environment. Following simulations, the set of chosen parameters was an initial population of 2 elements with a network degree of 2.

Note that all the networks used both in simulations and with real data rely on a low number of nodes, 30. A final remark is on the complexity of the algorithm: the fitness evaluation required to build the new population is linear in the number of nodes, like all GA, while the number of matings is equal to the number of links that, for the considered topologies, is again proportional to the number of nodes.

C. Real Experiments

Real experiments have been executed on the mobile platform ATRV-Jr manufactured by iRobot. It is a skid steering vehicle mainly designed to operate in outdoor environments. The ATRV-Jr has 4 wheels differentially driven by 2 DC motors: the motion is achieved by a differential thrust on the wheel pairs at the opposite sides. The mobile robot is equipped with two encoders, 17 sonar rangefinders, a laser

Algorithm 2: The proposed SSGA - iteration k

```
Data: Population of size $n$: $\{V = \{p_{j,k}\}, M\}$, $f(\cdot)$,
      $e(\cdot)$
Result: $V = \{p_{j,k+1}\}$

/* Average Fitness Evaluation                    */
$f_{aver} = \sum_{i=1}^{n} f(p_{i,k})/n$
/* Incidence Matrix Selection                    */
for $i$=1 to $n$ do
    for $j$=$i$ to $n$ do
        if $M(i,j) = 1$ then
            switch $Compare(\{f(p_{i,k}), f(p_{j,k})\}, f_{aver})$
            do
                case High-High
                    if $f(p_{i,k}) > f(p_{j,k})$ then
                        $p_{j,k} = Crossover(p_{i,k}, p_{j,k})$
                    else
                        $p_{i,k} = Crossover(p_{i,k}, p_{j,k})$
                    end
                case High-Low
                    $p_{j,k} = Mutation(p_{i,k})$
                case Low-High
                    $p_{i,k} = Mutation(p_{j,k})$
                case Low-Low
                    $p_{i,k} = Mutation(p_{i,k})$
                    $p_{j,k} = Mutation(p_{j,k})$
            end
        end
    end
end
$\{p_{j,k+1}\} = \{p_{j,k}\}$
/* Kidnap Detection                              */
if Kidnap Condition then
    Spreading Action
end
```

scanner (Sick LMS-220), an inertial platform (Crossbow DMU-6X), and a GPS receiver (Garmin GPS35-HVS). The sensory system is connected to the ATRV-Jr's on board PC (Pentium II, 350 MHz) running Linux, through serial port on a Rockeport multiserial port card. The robot is delivered with a software development environment called MOBILITY, which provides full access to the software servers available on the mobile platform. In particular, each server is assigned to control a specific hardware component (sensors and actuators). In this way all of them are reachable from the network exploiting a CORBA interface. However, it is important to underline that, although the robot is equipped with several sensors, only the encoders and the laser scanner (Sick LMS-220) have been exploited in this work.

Several analyses have been performed on real data to provide

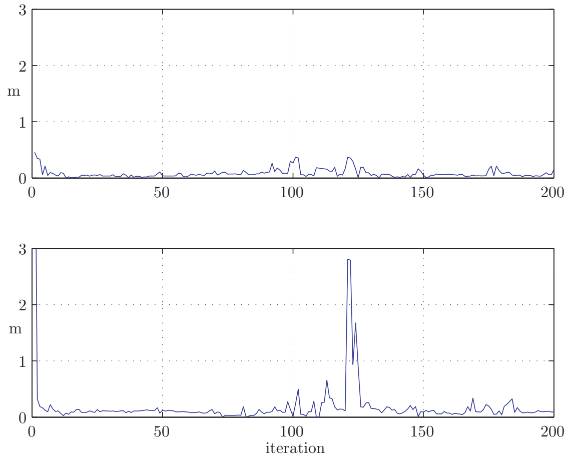

Fig. 3. Estimation errors along x (up) and y (down) axes

- a validation of the global localisation capability
- an evaluation of the tracking robustness
- a validation of the kidnap *sensing* strategy
- a comparison on real data of algorithm performance exploiting the *SW* and *SF* models

In Fig. 2 a first experiment is reported: the real robot, starting in S, moves downward and, after a U-turn around iteration 100, is kidnapped in K (iteration 120) to appear in N where it goes on until G. A video of the experiment has been downloaded with the paper and is also available at `www.dia.uniroma3.it/labrob/papers/icra07a/`. In the first 15 iterations, the network, randomly spread, forms some clusters, one of these will survive and after 20 iterations a single cluster remains as winner. Consequently, the tracking of the correct position is accomplished with very low errors (see Fig. 3). During the U-turn (iterations $95 - -110$) some measures, which do not correctly fit with the environmental model, along with the inaccuracy of the odometric prediction, will result in an inaccurate tracking. The fitness function is also affected by this situation, but the small increasing of the edge function shows how the network begins to explore new solutions in the surroundings to recover the error. Remember, the fitness function compares the estimated laser reads for a given hypothesis with the real data coming from the robot, whereas the edge function yields a measure of the similarity of the population.

The kidnap *sensing* strategy, as previously shown, relies on the use of the fitness function along with the edge function. In particular, an analysis has been carried out to find out the relationship between the kidnap event and the variation of these functions. According to the experimental results, when a kidnap occurs, the fitness function drastically decreases, whereas the edge function considerably increases because of the probabilistic operators effect. Consequently, a variation of the fitness function along with an increase of the edge function has been used as a threshold. In Fig. 3 (b) and (c) this situation is clearly represented: note that after the kidnap, the dispersion along y axis increases as long as the

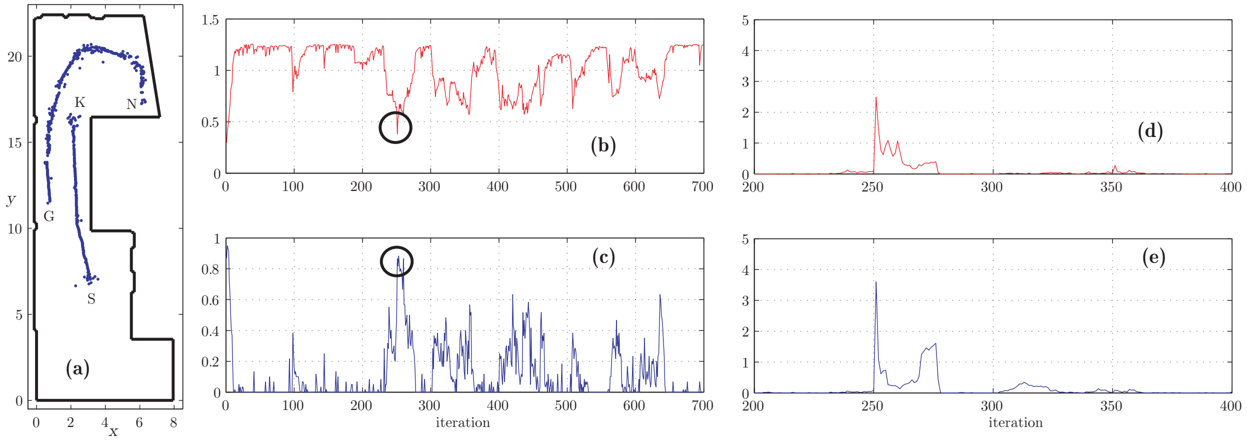

Fig. 4. (a) Map of the second environment and path of the robot (S:start, K:kidnap point; N:new start; E: end of path) - (b) Fitness function - (c) Edge function - (d) Dispersion along x axis - (e) Dispersion along y axis

error along y. In few iterations the error is recovered as the network starts to correctly track the robot.

A similar situation is shown in Fig. 4 (a second movie is available at the same address) where the robot, in a real environment, is kidnapped at iteration 250. A zoom on the iteration axes of Fig. 4 (d) and (e) has been performed to clearly show the behaviour of the dispersion during kidnap.

V. Conclusion

In this paper a spatially structured genetic algorithm for mobile robotic localisation has been proposed.

The novelty of the proposed algorithm is to take advantage of the complex network theory for the deployment of the population in order to more quickly find out the optimal solution. Moreover, it relies on the fact that the introduction of spatial structures in evolutionary algorithms helps to create evolutionary niches. As a consequence, being a niche a region in which a particular solution is preserved, a natural way to carry on multi-hypotheses is obtained.

Several experiments have been carried out in order to validate the strategy adopted to *sense* when a kidnap occurs as well as to prove the algorithm effectiveness. A comparison has been done to figure out the algorithm behaviour according to the Watts-Strogatz model and the scale-free Barabási-Albert model. Finally, a preliminary analysis has been performed to find out the relationship among the complex network properties and the algorithm performance.

According to the experimental results, the algorithm is able to solve both the global localisation problem and the kidnap problem, providing a pretty good tracking capability.

Several interesting challenges still remain for future work. A theoretical analysis to better figure out the improvement deriving from the use of the complex networks will be faced, such as an exploitation of the dynamic properties of the small-world model in case of self-organising networks. Finally, an implementation for a multirobot context, studying methodologies to fuse information coming from different networks, is currently under study.

References

[1] R. Kalman, "A new approach to linear filtering and prediction problems," *Transactions ASME Journal of Basic Engineering*, vol. 82, pp. 35–44, 1960.

[2] B. Schiele and J. L. Crowley, "A comparison of position estimation tecnhiques using occupancy grids," in *Proc. of the IEEE International Conference on Robotics and Automation*, 1994, pp. 1628–1634.

[3] W. Burgard, A. Derr, D. Fox, and A. B. Cremers, "Integrating global position estimation and position tracking for mobile robots: The dynamic markov localization approach," in *Proc. of the International Conference on Intelligent Robot and Systems*, 1998.

[4] A. Doucet, "On sequential simulation-based methods for bayesian filtering," CUED/F-INFENG/TR.310, Cambridge University, Department of Engineering, Signal Processing Group, Tech. Rep., 1998.

[5] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transaction on Signal Processing*, vol. 50, no. 2, February 2002.

[6] A. Gasparri, S. Panzieri, F. Pascucci, and G. Ulivi, "Genetic approach for a localisation problem based upon particle filters." in *Proc. of 8th Int. Symp. on Robot Control (SYROCO 2006)*, Bologna, Italy, 2006.

[7] D. Ashlock, M. Smucker, and J. Walker, "Graph based genetic algorithms," *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1362–1368, 1999.

[8] L. Moreno, J. M. Armingol, S. Garrido, A. D. L. Escalera, and M. A. Salichs, "A genetic algorithm for mobile robot localization using ultrasonic sensors," *J. Intell. Robotics Syst.*, vol. 34, no. 2, pp. 135–154, 2002.

[9] M. Annunziato, R. Huerta, M. Lucchetti, and L. S. Tsimring, "Artificial life optimization over complex networks," in *Proc. of the Fourth International ICSC Symposium on Engineering of Intelligent Systems*, Island of Madeira, Portugal, 2004.

[10] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks." *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.

[11] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, p. 509, 1999.

[12] G. M., P. M., and T. M., "Effects of scale-free and small-world topologies on binary coded self-adaptive cea," in *Lecture Notes in Computer Science*, J. Gottlieb and G. Raidl, Eds. Springer Verlag, 2006, pp. 86–98.

[13] E. P. and R. A., "On random graphs," *Publicationes Mathematicae*, vol. 6, pp. 290–297, 1959.

[14] X. F. Wang and G. Chen, "Complex networks: small-world, scale-free and beyond," *Circuits and Systems Magazine, IEEE*, vol. 3, no. 1, pp. 6–20, 2003.

[15] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.

[16] M. Giacobini, M. Tomassini, and A. Tettamanzi, "Takeover time curves in random and small-world structured populations," in *Proc. of the 2005 conference on Genetic and evolutionary computation*, 2005, pp. 1333 – 1340.