Evolving an Indoor Robotic Localization System Based on Wireless Networks

Gustavo Pessin[1,3], Fernando S. Osório[1], Jefferson R. Souza[1], Fausto G. Costa[1], Jó Ueyama[1], Denis F. Wolf[1], Torsten Braun[2], Patrícia A. Vargas[3]

[1] Institute of Mathematics and Computer Science (ICMC)
University of São Paulo (USP) - São Carlos, SP, Brazil
{pessin, fosorio, jrsouza, fausto, joueyama, denis}@icmc.usp.br
[2] Institute of Computer Science and Applied Mathematics
University of Bern - Bern, Switzerland
braun@iam.unibe.ch
[3] School of Mathematical and Computer Sciences (MACS)
Heriot-Watt University - Edinburgh, UK
p.a.vargas@hw.ac.uk

Abstract. This work addresses the evolution of an Artificial Neural Network (ANN) to assist in the problem of indoor robotic localization. We investigate the design and building of an autonomous localization system based on information gathered from Wireless Networks (WN). The paper focuses on the evolved ANN which provides the position of one robot in a space, as in a Cartesian plane, corroborating with the Evolutionary Robotic research area and showing its practical viability. The proposed system was tested on several experiments, evaluating not only the impact of different evolutionary computation parameters but also the role of the transfer functions on the evolution of the ANN. Results show that slight variations in the parameters lead to huge differences on the evolution process and therefore in the accuracy of the robot position.

1 Introduction

Mobile robot navigation is one of the most fundamental and challenging directions in mobile robot's research field and it has received great attention in recent years [6]. Intelligent navigation often depends on mapping schemes which turns out on depending on the localization scheme. For indoor or outdoor environments the mapping and localization schemes have their own features. Thereby, localization is a key problem in mobile robotics and it plays a pivotal role in various successful mobile robot systems [13].

This paper describes an investigation on the evolution of a system for indoor localization, which learns the position of one robot based on information gathered from WNs. The signal strength of several wireless nodes are used as input in the ANN in order to measure the position of one robot in an indoor space. The evolution of the ANN, detailed in section 2.1 is done using Particle Swarm Optimization [1, 3]. We show the complete hardware and software architecture for the robotic system developed so far. Our main focus in this work is to report findings which corroborate the use of evolutionary computation techniques to create autonomous intelligent robots [5].

In indoor spaces, sensors like lasers and cameras might be used for pose estimation [10], but they require landmarks (or maps) in the environment and a fair amount of computation to process complex algorithms. These sensors also have a limited field of vision, which makes harder the localization task. In the case of video cameras, the variation of light is also a serious issue. Another commonly used sensor is the encoder, which provides odometry. Odometry is an useful source of information in some cases [8] but it has an incremental error that usually invalidates its use in real systems.

Wireless Networks (WNs) are widely available in indoor environments and might allow efficient global localization while requiring relatively low computing resources. Other advantages of this technology are that it may provide high degrees of scalability and robustness. However, the inherent instability of the wireless signal does not allow it to be used directly for accurate position estimation. One machine learning technique that could reduce the instability of the signals of the WN is the Artificial Neural Networks; due to its capacity to learn from examples, as well as the generalization and adaptation of the outputs [9].

In [2], it has been shown that obtaining an absolute performance in localization, by means of a WN, depends on the environmental configuration. This means that different approaches are required for different environments, such as using different kinds of signals and filters. Evaluations in large indoor areas (like a building) present specific difficulties not always the same as in small indoor areas (like a room). These difficulties are related to the problem of attenuation and reflection of the signals on the walls and the different sources of interferences. The use of WNs addressing localization inside a building can be seen in [4, 7]. Another approach for localization uses Wireless Sensor Network (WSN) where a large number of small sensors are used to pick up information from the environment. The information acquired by the sensors can be regarded as a fingerprint [12]. The drawback of the canonical WSN approach is that, as it requires a huge number of resources, it could make the system more expensive.

This paper has the following structure: Section 2 outlines the methodology that is employed to set up and to evaluate the experiments. Section 3 describes all the evaluations that have been carried out. The final section makes some concluding comments and examines the future perspectives of this area of research.

2 Methodology

The indoor localization system uses an evolved ANN[4]. The inputs of the ANN are signals strength measurements from WNs (802.11b/g) received by the robot[5] from 8 statically positioned Access Points (AP) as shown in Fig. 1.

[4] Source-code and data files used to evolve the ANN are available in goo.gl/vfXN2

[5] Although we have used the humanoid robot NAO, the proposed methodology may be applied to any kind of device with wireless capabilities.

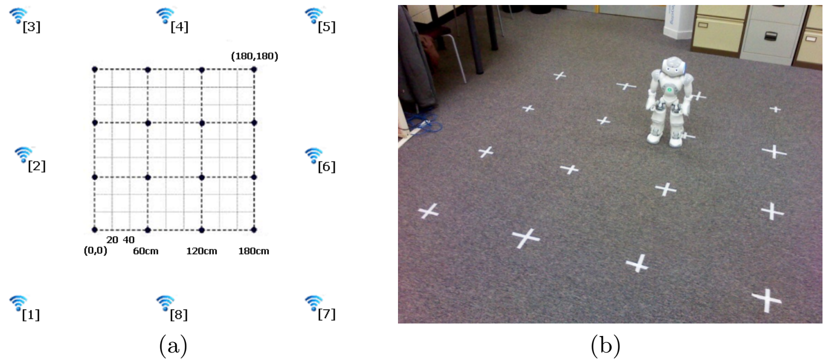


Fig. 1. (a) Graphical representation of the working area – It represents an area of 180 cm by 180 cm. (b) Picture of the working area with the robot, similar to what is represented in Figure 1(a). Each small cross are placed 60 cm long.

The evolution of the ANN is carried out using data collected by the robot. We use the robot inside the working area (Fig. 1(b)) and collected 3 minutes readings (i.e. ≈180 readings) at each marked point. With a displacement of 60 cm, mapping out a plane of 180 cm by 180 cm, it means 16 points to read resulting in ≈2880 readings altogether.

The signal obtained from the WN is the Received Signal Strength Indication (RSSI). This value is obtained with the aid of the GNU/Linux command *iwlist* (used as *iwlist <interface> scanning*). As we use the *iwlist* command, there is no need to establish a connection (or login) with different specific networks. The scan of the networks, without a connection, provides enough information for this evaluation. Without a connection, the system becomes easier to use, more lightweight and flexible. Furthermore, as the robot NAO has an operating system based on GNU/Linux, this approach may be generalized to any other GNU/Linux based system.

Our approach relies on the ANN learning and generalization capabilities in an attempt to reduce the effect of unstable data (due to signal strength oscillation), and increase the accuracy of the position estimation of the robot. However, as the values obtained from the reading of the APs are quite unstable, we improve the learning capability using the noise filter proposed in [11]. The behaviour of the noise filter can seen in Fig. 2(a), where two lines represent scanning of one AP in a period of time. The red line shows the raw value and the black line shows how the median filter removes some of the noise. Although it generates a delay of 8 seconds in acquiring the new position it was shown in [11] that the accuracy turns out to be widely better.

The number of neurons in the input layer is equivalent to the number of available APs – as we use 8 APs, the inputs of the ANN use one neuron for each network signal. The order is important, and hence, AP 1 was linked to neuron 1, AP 2 with neuron 2 and so on. The outputs of the network are two values, i.e. the coordinates (x, y). We measure the output errors by using the Euclidean

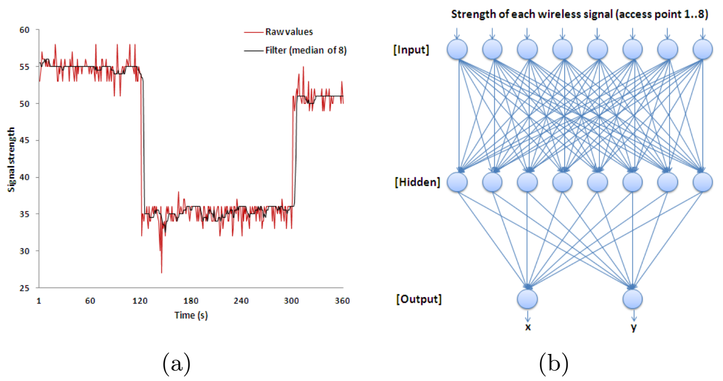


Fig. 2. (a) Sample of filter behaviour – The red line shows the raw value from one of the access points. The black line shows how the median filter removes some of the noise. (b) Example of ANN topology.

distance, as shown in Eq. 1. The value d is the error (distance, in centimetres), (x_1, y_1) are the expected values from the ANN validation set and (x_2, y_2) are the obtained value while using the ANN.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

2.1 Evolutionary Localization

As we seek to evaluate ANN characteristics and also the evolution process, we started with a simple ANN topology as it can be seen in Fig. 2(b). We evaluate changes in the ANN topology and in the role of the transfer function. Furthermore, as the evolution is carried out using Particle Swarm Optimization (PSO) [1, 3] we evaluate several aspects that influence the search efficiency in the PSO. These aspects are related to confidence models, neighbourhood topology, and inertia among others.

We use PSO to evolve two different structures. In the first one, we use the PSO to evolve just the ANN weights. In the second, we evolve the ANN weights plus the slope of the transfer function. As an example, the ANN in Fig. 2(b) has 80 connections plus 10 weights for bias, hence, the PSO particle has 90 positions (i.e. it is an vector with 90 positions). For the slope, we consider its use just in the hidden layer, and it is the same value for all neurons. Hence, it adds only one more value to the PSO particle. The evolutionary process considerers 2/3 of the dataset as training data and 1/3 as validation data. The ANNs are evolved for 10k cycles (generations) and all presented results are from the validation dataset.

The PSO is a stochastic optimization technique, inspired by social behaviour of bird flocking and fish schooling [1, 3]. The optimization process occurs in two different ways simultaneously: through cooperation (group learning) and competition (individual learning) among particles (individuals) from a swarm (population). It shares many concepts with evolutionary computation techniques such as Genetic Algorithms (GA), where there is an initial population (where each individual represents a possible solution) and a fitness function (whose value represents how far an individual is to an expected solution). However, unlike GA, PSO has no explicit concepts of evolution operators such as crossover or mutation. In the PSO, there is a swarm of randomly created particles. On each algorithm iteration, each particle is updated following: (i) the best population fitness; (ii) the best fitness found by the particle (considering past generations of the particle). Each particle has a position x (or a position vector) and a velocity v (or velocity vector). The position represents a solution for the problem and the velocity defines the particles displacement direction weight.

New particle's position is given by Eq. 2. Where x_k^i is the position of particle i at instant k and v_k^i is particle's i velocity at k moment. Particle's velocity is updated according to Eq. 3.

$$x_{k+1}^i = x_k^i + v_{k+1}^i \tag{2}$$

$$v_{k+1}^i = w \cdot v_k^i + c_1 \cdot r_1(pbest - x_k^i) + c_2 \cdot r_2(gbest - x_k^i) \tag{3}$$

On Eq. 3, v_k^i is the particle actual velocity, w represents a particle inertia parameter, *pbest* is the best position among all positions found by the individual (particle best), *gbest* is the best position among all positions found by the group (group best), c_1 and c_2 are trust parameters, r_1 and r_2 are random numbers between 0 and 1. Parameters (w, c_1, c_2, r_1 e r_2) are detailed below.

The velocity is the optimization's process guide parameter [3] and reflects both particle's individual knowledge and group knowledge. Individual knowledge is known as *Cognitive Component* while group knowledge is known as *Social Component*. Velocity consists of a three-term sum: (i) Previous speed: utilized as a displacement direction memory and can be seen as a parameter that avoids drastic direction changes; (ii) Cognitive Component: directs the individual to their best position found so far (i.e. memory of the particle); (iii) Social Component: directs the individual to the best particle in the group.

Parameters c_1 and c_2 (confidence or trust) are used to define individual or social tendency importance. Default PSO works with static and equal trust values (c_1=c_2), which means that the group experience and the individual experience are equally important (called *Full Model*). When parameter c_1 is zero and parameter c_2 is higher than zero, PSO uses only group information (called *Social Model*). When parameter c_2 is zero and parameter c_1 is higher than zero, PSO uses only particle's information, disregarding group experience (called *Cognitive Model*). Random value introduction (r_1 and r_2) on velocity adjustment allows PSO to explore on a better way the search space [3]. Inertia parameter aims to balance local or global search. As the value approximates to 1.0, search gets close to global while lower values allow local search. Usually this value is between 0.4 and 0.9. Some authors suggest its linear decay, but they warn that it is not always the best solution. Most parameters are problem-dependent [3, 14].

3 Results

This section describes the four steps that were carried out to perform the evaluations, considering changes in the ANN and PSO techniques. As a first step we sought to evaluate the impact of using different range values in the initialization of PSO's velocity and position. Furthermore we evaluate how the number of generations and the swarm size impact the evolution.

Table 1(a) shows the evaluation set related to swarm size and the number of generations. Table 1(b) shows the evaluation set for weights of connections (i.e. position) and velocity parameters in the PSO.

Table 1. First evaluation set performed with the PSO.

(a)

Generations	Swarm Size		
	200	500	1000
500	E1	E2	E3
1000	E4	E5	E6
2000	E7	E8	E9

(b)

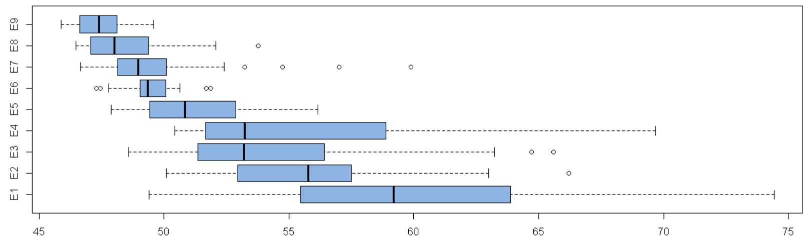
Velocity	Position		
	{-2.0;2.0}	{-5.0;5.0}	{-20.0;20.0}
{-2.0;2.0}	A1	A2	A3
{-5.0;5.0}	A4	A5	A6
{-20.0;20.0}	A7	A8	A9

We performed 25 runs for each parameter set, considering different random seeds on each initialization. The results can be seen in Fig. 3. We can see in Fig. 3(a) that more generations and bigger swarm size provides better results (E9). However, even using swarm size equal to 1,000 and number of generations equal to 2,000 the evolution process was not reaching learning stabilization (the learning curve still have slightly improvements).

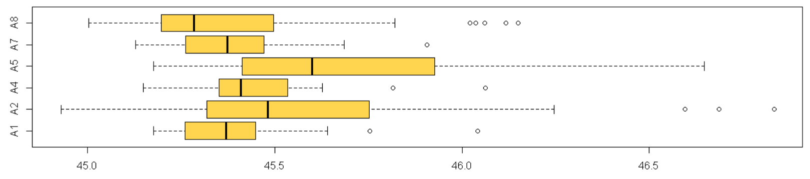
In Fig. 3(b) we can see that the two sets that obtained the best (lower error) results considering the range of position and velocity are A8 and A2. Both use positions between {-5.0;5.0} but have different velocities range. We can see that although A8 has the lowest minimum error it has the biggest dispersion among all. The A2 set has the lowest median and a minimum error close to A8. Although, considering the graph scale (2 cm), all results are not much different. Thereafter, to the next steps, we maintain the set A2 in the PSO. However, given that the evolution process was not reaching complete stabilization, we extrapolate E9 with 10k generations instead of 2k.

As a second step, we wanted to understand the behaviour of the PSO evolving the ANN considering confidence models, the inertia and the role of the transfer function. Table 2 shows the evaluated parameter set. Linear and Logistic are the two types of transfer function used in the ANN hidden layer.

We can see that results (Fig. 4) using the *Cognitive Model* or the *Social Model* were worse than using the *Full Model.* The four best sets are {Fi3, Fi5, Fo3, Fo5}. We can see that the sets with Logistic Transfer Function obtained best results near to 15 cm while the sets with Linear Transfer Function obtained best results near to 45 cm. In these sets, we can also see that when inertia was used as 0.3 we have better results than using inertia equal to 0.5 or 0.7. It makes sense due to the fact that lower the inertia better the local search (fine tuning). Hence, the best parameter set which we maintain to next steps is Fo3.



(a)



(b)

Fig. 3. (a) Results using different swarm size and number of generation (Table 1(a)). (b) Results using different initialization in PSO position and velocity (Table 1(b)). The graphs are in different scales. The x axes represent the error in centimetres.

Table 2. Second evaluation set performed with the PSO (inertia and confidence models) and the ANN (transfer function).

Inertia	Full		Social		Cognitive	
	Linear	Logistic	Linear	Logistic	Linear	Logistic
0.3	Fi3	Fo3	Si3	So3	Ci3	Co3
0.5	Fi5	Fo5	Si5	So5	Ci5	Co5
0.7	Fi7	Fo7	Si7	So7	Ci7	Co7

Fo3 set uses Logistic Transfer Function with a slope of 0.02. We also had a different PSO evolving the slope, however, the results when we leave the PSO to evolve the slope were similar to the use of the pre-defined value. Of course, the finding of slope equal to 0.02 was not trivial as it was encountered analysing the output of the sum of the hidden layers. Such a situation may encourage the use of the PSO in order to find the slope of the Transfer Function. We also performed some evaluations taking into account linear decay of the inertia value, but the results were not better than the currently configuration.

In the third step, we sought to evaluate the impact of using different PSO neighbourhood topologies (NT). PSO NT are related to the choice of the best particle to follow. In the star model, all particles follow the best among all. Nevertheless, it may sometimes be more susceptible to local minima. Others NT may be less susceptible to local minima, where the particle does not to follow the global best but the best of some subpopulation. It, in general, also increases the diversity. We have evaluated 4 different types of NT: (i) the star model; (ii) two subpopulations – i.e. 2 groups of neighbours; (iii) four subpopulations – i.e. 4 groups of neighbours; and (iv) four subpopulations but being the best particle to follow the average among the two best in the subpopulations. Upon these results

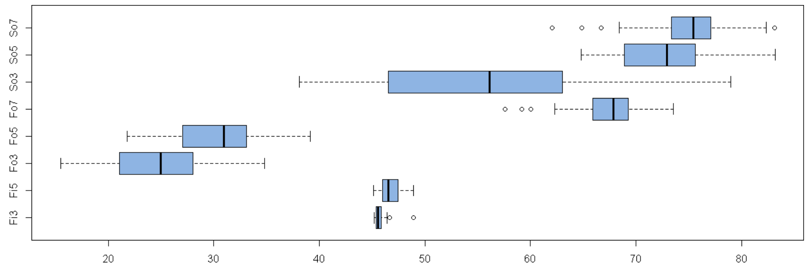


Fig. 4. Results using different PSO confidence models and inertia, as show in Table 2. We show the sets where the average error were lower than 80 cm – 8 of 18 sets. The x axis represents the error in centimetres.

we performed a statistical test to verify its significance. Using t-test among all sets they are accepted as equivalent (using 95% of confidence). It means that, for this problem, the use of different NT does not substantially affect the results.

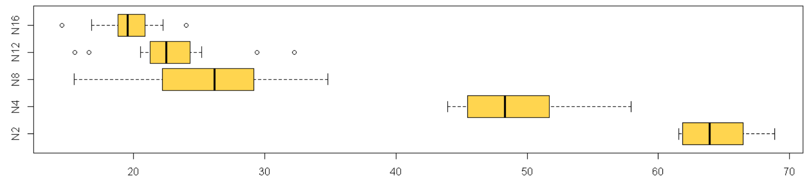


Fig. 5. Results using different number of neurons in the ANN hidden layer. The graphs are in different scales. The x axes represent the error in centimetres.

The final step was the evaluation of the number of neurons in the hidden layer. We evaluate the use of 2, 4, 8, 12 and 16 neurons. Results can be seen in Fig. 5. We can see that using 2 and 4 neurons, the results are quite bad. Using 8 and 12 neurons they have good minimum errors but high dispersion. The ANN with 16 neurons presents the lowest error and the more homogeneous results. Statistical analyses (t-test) upon N12 and N16 shows that, with 95% of confidence, they are not accepted as equivalent (p-value of 0.013), i.e., the use of 16 neurons does improve the system.

After the evaluation of the number of neurons in the hidden layer, we performed a new evaluation considering the best set found so far but running the evolutionary process for 100k generations instead of 10k. Results can be seen in Fig. 6. The data shown in Fig. 6(a) and 6(b) presents an average error and standard deviation, respectively, of (14.6, 18.4) and (10.2, 14.4), i.e. running the PSO for 100k generations allowed us to decrease the average error in more $\approx$30%. We can see that Fig. 6(a) has less results in the first class (0 to 5 cm) and a bigger tail than Fig. 6(b). Statistical evaluation using the Mann-Whitney test (as it is a non normal distribution) showed p-value equal to $1.517e^{-11}$ (not

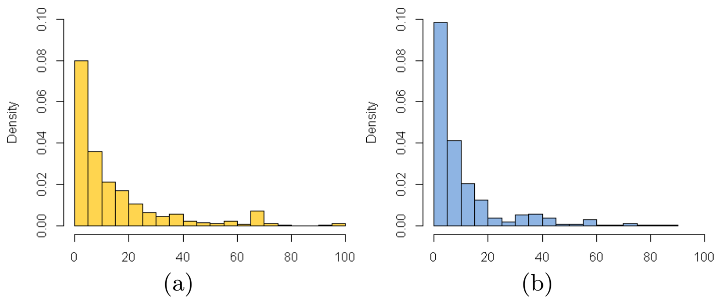


Fig. 6. Histogram of the localization error using the best acquired ANN. (a) Using 10k generation in the PSO. (b) Using 100k generation in the PSO. The x axes represent the error in centimetres.

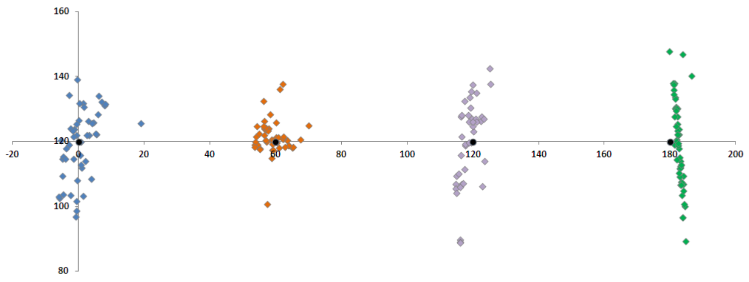


Fig. 7. Section of the plane (Fig. 1(a)) with expected and obtained values for 4 coordinates. The black dots are the expected value; diamonds show the obtained values. Each colour represents a different position. Axes x and y are in centimetres.

accepted as equivalent using 95% of confidence). Fig. 7 shows a section of the plane (Fig. 1(a)) with expected and obtained values for 4 positions, using the best acquired ANN. For all positions in the evaluated plane, 86% of the errors are below 20 cm.

4 Conclusion and Future Work

In this paper we have shown an investigation addressing the evolution and the use of an ANN to assist in the problem of indoor localization by using data gathered from WNs. Upon the data obtained from the WN we employed a median noise filter as shown in [11]. We evaluated several PSO and ANN settings. Results showed that the use of transfer function in the ANN along with the PSO Full model allowed us to decrease the average error from ≈45 cm to ≈15 cm. Also, we might see that the use PSO neighbourhood topology did not allow us to have any significant improvement. Finally, the system could be improved using larger number of neurons in the hidden layer and larger number of generations, leading to an average error of ≈10 cm.

Nevertheless, it is important to notice that results presented in this paper cannot be directly compared with results from [11] because the papers have not employed the same data, as they were collect in different environment and with different robots. Future work may include an investigation to improve the local search, since even using a huge number of generations we still have a slightly decreasing error. Further, we are considering the other two approaches, that is, the comparison with others Evolutionary Techniques and a comparison with classical ANN learning algorithms to verify its accuracy and efficiency.

5 Acknowledgments

The authors would like to acknowledge the financial support granted by CNPq and FAPESP to the INCT-SEC (National Institute of Science and Technology – Critical Embedded Systems – Brazil), processes ID 573963/2008-8 and 08/57870-9. Also, we would like to acknowledge the Capes Foundation, Ministry of Education of Brazil, process BEX 4202-11-2.

References

1. Eberhart, R.C., Kennedy, J., Shi, Y.: Swarm Intelligence. M. Kaufmann (2001)
2. Elnahrawy, E., Li, X., Martin, R.: The limits of localization using signal strength: a comparative study. In: IEEE SECON. pp. 406–414 (2004)
3. Engelbrecht, A.P.: Fundamentals of Comp. Swarm Intelligence. Wiley (2005)
4. Espinace, P., Soto, A., Torres-Torriti, M.: Real-time robot localization in indoor environments using structural information. IEEE LARS (2008)
5. Fogel, D.: Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press Series on Computational Intelligence (2006)
6. Fu, S., Hou, Z., Yang, G.: An indoor navigation system for autonomous mobile robot using wsn. In: Networking, Sensing and Control. pp. 227–232 (2009)
7. Ladd, A., Bekris, K., Rudys, A., Wallach, D., Kavraki, L.: On the feasibility of using wireless ethernet for indoor localization. Robotics and Automation, IEEE Trans. on 20(3), 555 – 559 (2004)
8. Martinelli, A.: The odometry error of a mobile robot with a synchronous drive system. Robotics and Automation, IEEE Trans. on 18(3), 399–405 (2002)
9. Mitchell, T.M.: Machine Learning. McGraw-Hill (1997)
10. Napier, A., Sibley, G., Newman, P.: Real-time bounded-error pose estimation for road vehicles using vision. In: IEEE Conf. on Intelligent Transp. Systems (2010)
11. Pessin, G., Osório, F.S., Ueyama, J., Souza, J.R., Wolf, D.F., Braun, T., Vargas, P.A.: Evaluating the impact of the number of access points in mobile robots localization using artificial neural networks. In: Proc. of the 5th International Conference on Communication System Software and Middleware. pp. 10:1–10:9 (2011)
12. Robles, J., Deicke, M., Lehnert, R.: 3d fingerprint-based localization for wireless sensor networks. In: Positioning Navigation and Communication (WPNC) (2010)
13. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust monte carlo localization for mobile robots. Artificial intelligence 128(1-2), 99–141 (2001)
14. Yao, X.: Evolving artificial neural networks. Proc. of IEEE 87(9), 1423–1447 (1999)