Investigation on the Evolution of an Indoor Robotic Localization System Based on Wireless Networks

Gustavo Pessin[a,c], Fernando S. Osório[a], Jefferson R. Souza[a], Jó Ueyama[a], Fausto G. Costa[a], Denis F. Wolf[a], Desislava Dimitrova[b], Torsten Braun[b], Patrícia A. Vargas[c]

[a] *University of São Paulo (USP)*
Institute of Mathematics and Computer Science (ICMC)
São Carlos, São Paulo, Brazil
{pessin, fosorio, jrsouza, joueyama, fausto, denis}@icmc.usp.br
[b] *University of Bern*
Institute of Computer Science and Applied Mathematics
Bern, Switzerland
{dimitrova, braun}@iam.unibe.ch
[c] *Heriot-Watt University*
Institute of Computer Science and Applied Mathematics
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 a robot in a space, as in a Cartesian coordinate system, corroborating with the Evolutionary Robotic research area and showing its practical viability. The proposed system was tested in 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.

Keywords: Indoor Robotic Localization, Evolutionary Robotics, Artificial Neural Network, Particle Swarm Optimization

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 (Fu et al., 2009). Intelligent navigation often depends on mapping schemes which turns out on depending on the localization scheme. For either 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 (Thrun et al., 2001).

This paper describes an investigation on the evolution of a system for indoor localization, which estimates 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 determine 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 (Eberhart et al., 2001; Engelbrecht, 2005). 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 (Fogel, 2006). This paper is an extended version of our previous work (Pessin et al., 2012). It also aims to describe in a more thorough way the applied methodology, the experiments and the discussion about the obtained results. The main novelty aspect in this paper is the investigation on several new parameters of the PSO.

In indoor spaces, sensors like laser range finders and cameras might be used for pose estimation (Napier et al., 2010), 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 (Martinelli, 2002) 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 (Mitchell, 1997).

In (Elnahrawy et al., 2004), 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 (Espinace et al., 2008; Ladd et al., 2004). 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 (Robles et al., 2010). The drawback of the canonical WSN approach is that, as it requires a huge number of resources, it could increase the time to collect the fingerprint and it also often lacks of flexibility and scalability.

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.

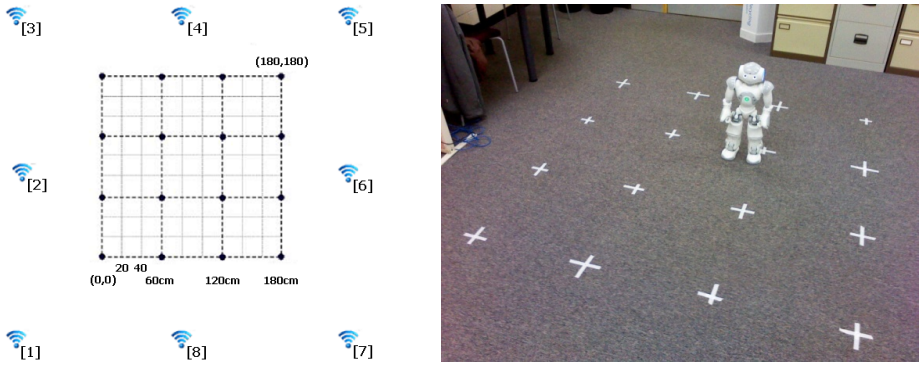
2. Methodology

The indoor localization system uses an evolved ANN[1]. The inputs of the ANN are signals strength measurements from WNs (802.11b/g) received by the robot[2] from 8 statically positioned Access Points (AP) as shown in Fig. 1(a) and 1(b). 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*[3]. As we have used the *iwlist* command, there were no needs to establish a connection (or login) with different specific networks. The scan of the networks,

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

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

[3] Used as *iwlist <interface> scanning*



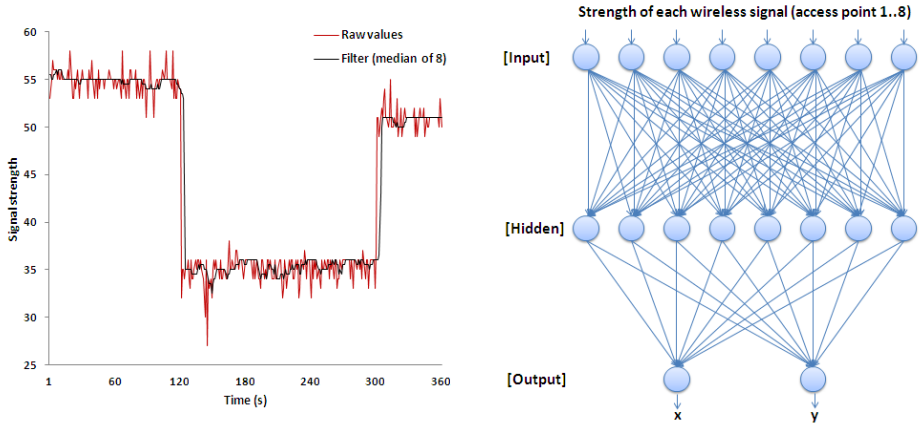
(a) (b)

Figure 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.

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.

The evolution of the ANN is carried out using data collected by the robot. We use the robot within the working area (Fig. 1(b)) and collected readings over 3 minutes (i.e. $\approx$ 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 $\approx$ 2880 readings altogether.

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 (Pessin et al., 2011). The



(a) (b)

Figure 2: (a) Sample of filter behavior – 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.

behavior of the noise filter can be seen in Fig. 2(a), where two lines represent scanning of one AP over 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 (Pessin et al., 2011) 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 distance, as shown in Eq. 1. The value d is the error (distance, in centimeters), (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) (Eberhart et al., 2001; Engelbrecht, 2005) 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 PSO is a stochastic optimization technique, inspired by social behavior of bird flocking and fish schooling (Eberhart et al., 2001; Engelbrecht, 2005). The optimization process occurs in two different ways simultaneously: through co-operation (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. At 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 (Engelbrecht, 2005) 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 (Engelbrecht, 2005). 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 (Engelbrecht, 2005; Yao, 1999).

3. Results

This section describes the six evaluations we have performed, considering several changes in the ANN and PSO techniques. Results are always presented from the validation dataset (Optimum Generalization Point). We performed 25 runs for each parameter set, considering different random seeds on each initialization. The evolutionary process considers 2/3 of the dataset as training data and 1/3 as validation data.

3.1. Range of Velocity and Position

Velocity is the optimization's process guide parameter (Engelbrecht, 2005) and reflects both particle's individual knowledge and group knowledge. In mathematical terms, it specifies the weight of the displacement vector. Higher velocity leads to larger displacement in the position and it is an issue related to global and local search (Pant et al., 2009; Herrera and Zhang, 2009). Position, in this work, is regarded as the synaptic weight of the connections.

We evaluate the impact of using different range values in the initialization of PSO's velocity and position. Furthermore we also evaluate how the number of generations and the swarm size impact the evolution. Table 1 shows the evaluation set related to swarm size and the number of generations. Table 2 shows the evaluation set for weights of connections (i.e. position) and velocity parameters in the PSO. 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

Table 1: Evaluation set used to investigate the swarm size and the number of generations in the PSO.

Generations	Swarm Size		
	200	500	1,000
500	E1	E2	E3
1,000	E4	E5	E6
2,000	E7	E8	E9

Table 2: Evaluation set used to investigate the weights of connections (i.e. position) and velocity parameters in the PSO.

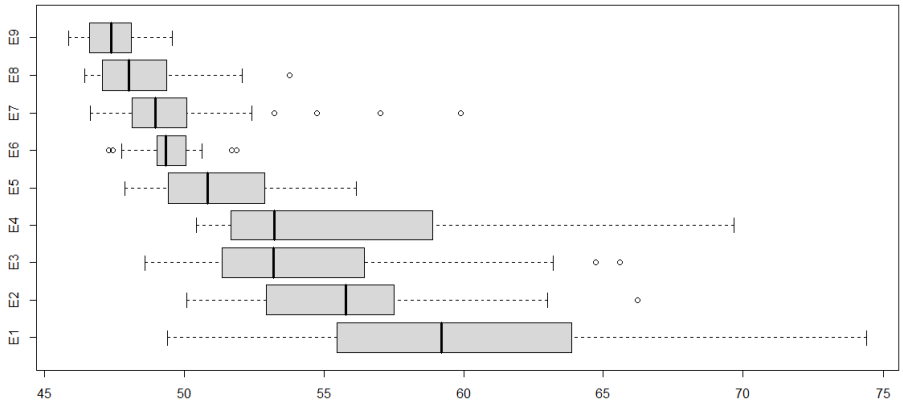
Velocity	Position		
	{-2.0;2.0}	{-5.0;5.0}	{-20.0;20.0}
{-2.0;2.0}	A1	A2	A3
{-5.0;5.0}	A4	A5	A6
{-20.0;20.0}	A7	A8	A9

of generations equal to 2,000 the evolution process was not reaching learning stabilization (the learning curve still had slight improvements).

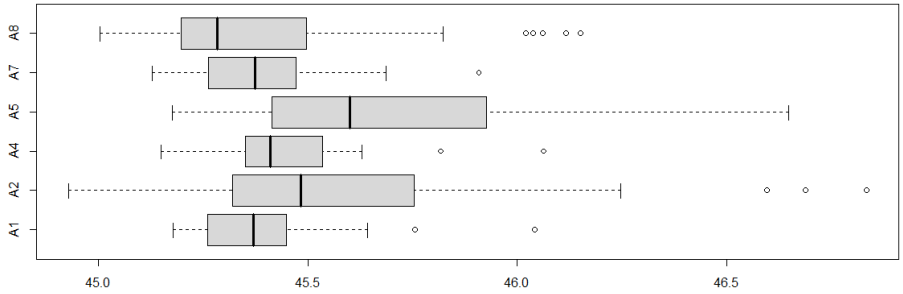
In Fig. 3(b) we can see that the two sets that obtained the best (lower error) results considering the range of velocity and position are A8 and A2. Both use positions between {-5.0;5.0} but have different velocities range. We can see that although A2 has the lowest minimum error it has the biggest dispersion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

3.2. Confidence models, inertia and the role of the transfer function

The confidence model is an issue directly related to how one particle is influenced by another particle of the swarm or by itself. It is extremely related



(a)



(b)

Figure 3: (a) Results using different swarm size and number of generation (an shown in Table 1). We can see that more generations and bigger swarm size provides better results (E9). (b) Results using different initialization in PSO velocity and position (an shown in Table 2). Best (lower error) can be seen in A8 and A2. The graphs are in different scales. The x axes represent the error in centimeters. Sets A3, A6 and A9 are not presented in the figure due to have average errors larger than 55 cm.

to the diversity of the population. Greater diversity may lead to better global search while small diversity may lead to premature convergence (Ozcan and Mohan, 1999; Kennedy, 1997).

The inertia, as presented in the Section 2.1, 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. The inertia weight (w) controls the momentum of the particle by weighting the previous velocity (Malik et al., 2007).

Transfer functions are related to the production of a scalar neuron output (Dorofki et al., 2012; Haykin, 2008). Usually, a transfer function (or activation function) is used to limit the output of a neuron or to smooth the output. It is related to the learning and generalization capabilities of the ANN.

We seek to understand the behavior of the PSO evolving the ANN considering confidence models, the inertia and the role of the transfer function. Table 3 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 the lower the inertia the better the local search (fine tuning). Hence, the best parameter set which we maintain to next steps is Fo3.

Table 3: Evaluation set used to investigate the inertia and confidence models (in the PSO) and the transfer function (in the ANN).

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

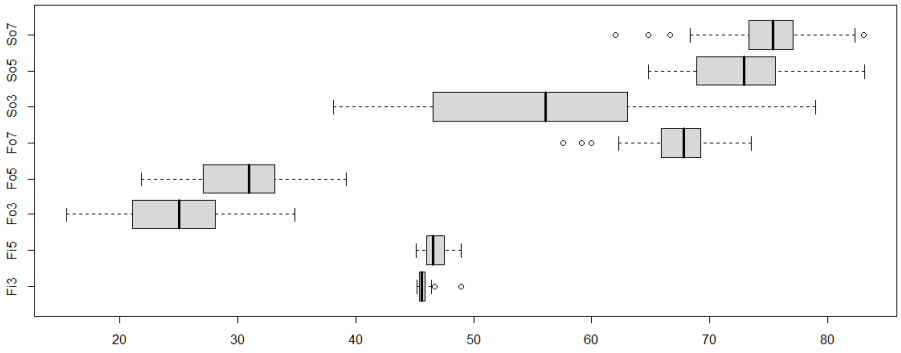


Figure 4: Results using different PSO confidence models and inertia, as show in Table 3. We show the sets where the average error were lower than 80 cm – 8 of 18 sets. The x axis represents the error in centimeters. The best set (lowest median) is Fo3 (PSO full model, Transfer function logistic and inertia equal to 0.3).

We have used 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 analyzing 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. Some evaluations taking into account non-linear inertia will be show in Section 3.5.

3.3. Number of Neurons in the Hidden Layer

The number of neurons in the ANN hidden layer is related to the learning and generalization capabilities. Too few neurons can lead to underfitting (lack of learning) while too many neurons can lead to overfitting (learning too well the training data and lacking generalization capabilities) and waste of computational resources (Teoh et al., 2006; Huang, 2003).

We evaluate different numbers of neurons in the hidden layer to understand the learning and generalization capabilities. Hence, we evaluate the use of 2, 4, 8, 12 and 16 neurons. Results can be seen in Fig. 5. We can see poorest results using 2 and 4 neurons. Using 8 and 12 neurons leads to good minimum errors but high dispersion. The ANN with 16 neurons presents the lowest error and the more homogeneous results. Statistical analyses (Welch Two Sample t-test[4]) upon N12 and N16 shows that, with 95% of confidence, they are not accepted as equivalent

[4]N12 and N16 are accepted as normal distribution using the Shapiro-Wilk normality test, p-values equal to 0.634 and 0.928.

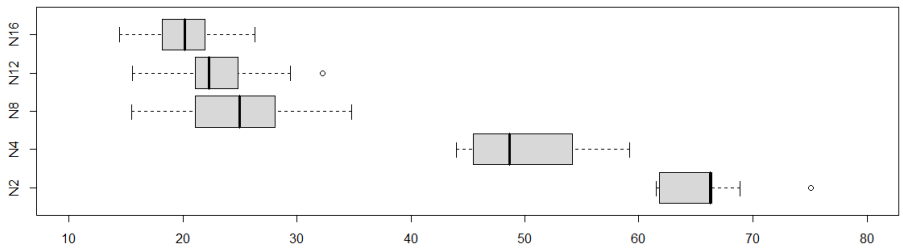


Figure 5: Results using different number of neurons in the ANN hidden layer. The *x* axis represent the error in centimeters. We can see poorest results using 2 and 4 neurons. Using 8 and 12 neurons they have good minimum errors but high dispersion. The ANN with 16 neurons presents the lowest error and the more homogeneous results, being $\approx 12\%$ better than N12.

(p-value of 0.011), i.e., the use of 16 neurons does improve the system accuracy in $\approx 12\%$.

3.4. Neighbourhood Topologies

Neighbourhood topologies are related to the choice of the best particle to follow. In the star model, all particles follow the best among all (example in Fig. 6(a)). Nevertheless, it may sometimes be more susceptible to local minima. Others neighbourhood topologies may be less susceptible to local minima, where the particle does not to follow the global best but the best of some subpopulation (example in Fig. 6(b)). It, in general, could increase the diversity. We have evaluated 7 different types of neighbourhood topologies:

- The star model (ST). As shown in Figure 6(a) all the particles (light gray) follow the best particle of all (light orange).
- Two subpopulations – i.e. 2 groups of neighbours having two *gbests*, one for each subpopulation (S2). As shown in Figure 6(b) the particles (light gray) follow the best particle of its neighbourhood (light orange).

- Four subpopulations – i.e. 4 groups of neighbours having four *gbests*, one for each subpopulation (S4).

- Two subpopulations having one *gbest* which is the average of the *gbest* of each subpopulation (AV2). As shown in Figure 6(c) the particles (light gray) follow the average of the best particle of each neighbourhood (light green).

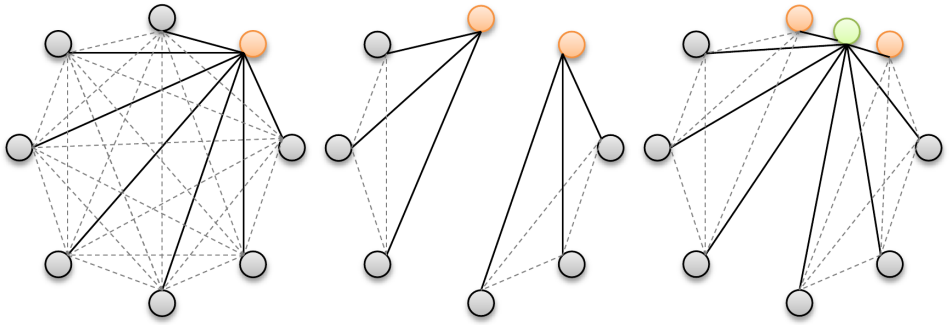
- Four subpopulations having one *gbest* which is the average of the *gbest* of each subpopulation (AV4).

- Two subpopulations where the particle follow one or another position of the *gbest* randomly – i.e. following positions of the particle, something like crossover (RPR).

- Two subpopulations where the particle follow one or another *gbest* randomly – i.e. following all the particles (RPO).

We can see in Fig. 7 the results of using different types of neighbourhood topologies. We can see the poorest results in RPO, RPR and AV4. ST, S2, S4 and AV2 were evaluated statistically and showed p-value from the Kruskal-Wallis rank sum test[5] equal to 0.355 – i.e. the sets are accepted as equivalent using 95% of confidence. If the sets are accepted as equivalent, we ought use the least complex option, which is the Star Model (ST).

[5]We have used Kruskal-Wallis rank sum test because AV2 is not accepted as a normal distribution.



(a) (b) (c)

Figure 6: Examples of neighbourhood topologies. (a) Star mode – every particle may see any other particles (dashed line), although they follow (straight line) just the global best (light orange). (b) Two subpopulations – i.e. 2 groups of neighbours having two *gbests*, one for each subpopulation. Every particle may see any other particles (dashed line) in their subpopulation, although they follow (straight line) the global best (light orange) of each subpopulation. (c) Altered two subpopulations – there are two subpopulations and the *gbest* is the average of the *gbest* of each subpopulation. The particles (light gray) follow (straight line) the average of the best particle of each neighbourhood (light green).

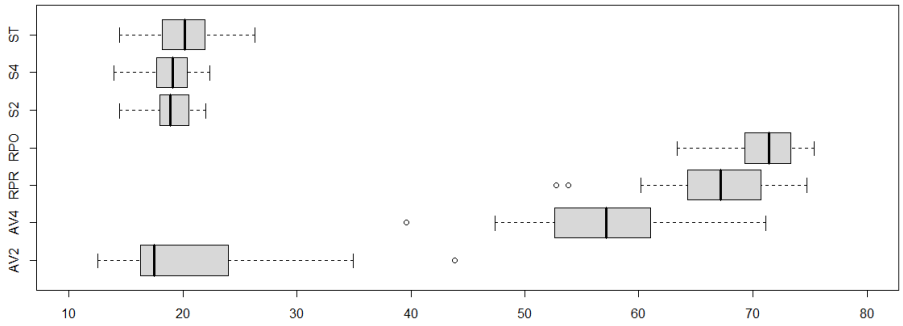


Figure 7: Results using different types of neighbourhood topologies. We can see the poorest results in RPO, RPR and AV4. ST, S2, S4 and AV2 were evaluated statistically and showed p-value from the Kruskal-Wallis rank sum test equal to 0.355 – i.e. these 4 sets are accepted as equivalent using 95% of confidence.

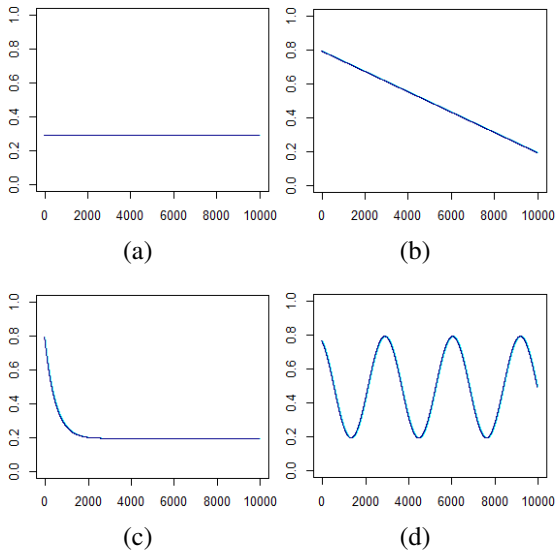


Figure 8: Example of inertia behaviour: (a) Uniform ($w = 0.3$). (b) Linear decay. (c) Exponential. (d) Sinusoid. Values are normalized between 0.8 and 0.2.

3.5. Non-uniform Inertia

In previous Sections 2.1 and 3.2 we have presented some explanations about the inertia influence. We have shown that maintaining the inertia uniform, equal to 0.3 have had better results than inertia equal to 0.5 and 0.7.

Works (Silveira et al., 2010; Deep et al., 2011; Kentzoglanakis and Poole, 2009) have evaluated non-uniform inertia showing that sometimes it may improve the PSO. Hence, this section aims to evaluate the impact of using non-uniform inertia in the localization problem. We have evaluated 4 different types of inertia behaviour, the uniform one (as Section 3.2 – Figure 8(a)) and three non-uniform: (i) linear decay (Figure 8(b)), (ii) exponential decay (Figure 8(c)) and (iii) sinusoid (Figure 8(d)).

Figure 9 shows the result of the evolution considering the four evaluated types

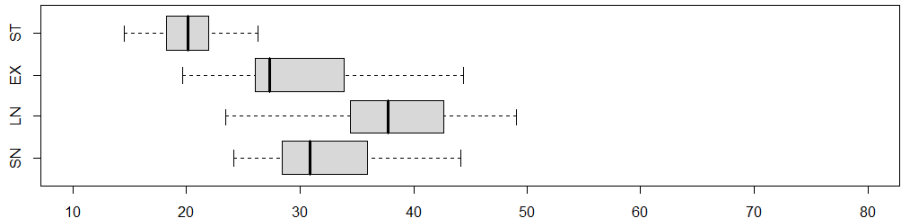


Figure 9: Result of the evolution considering the four evaluated types of inertia behaviour. We can see that the use of uniform inertia has presented better results than non-uniform ones.

of inertia behaviour. We can see that the use of uniform inertia has presented better results than non-uniform ones. The result of the statistical test (Welch Two Sample t-test) showed p-values lower than 0.000 for the sets {(ST, EX), (ST, LN), (ST, SN)}, i.e. They are not accepted as equivalent using 95% of confidence. Hence, for this problem, we can see that the use of uniform inertia showed significant better results than other types of inertia.

3.6. More Generations

As a last step, 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. 10. The data shown in Fig. 10(a) and 10(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. 10(a) has less results in the first class (0 to 5 cm) and a bigger tail than Fig. 10(b). Statistical evaluation using the Mann-Whitney test (as it is a non-normal distribution) showed p-value equal to $1.517e^{-11}$ (not accepted as equivalent using 95% of confidence). Fig. 11 shows a section of the plane (Fig. 1(a)) with expected and obtained values for 4 positions, using the

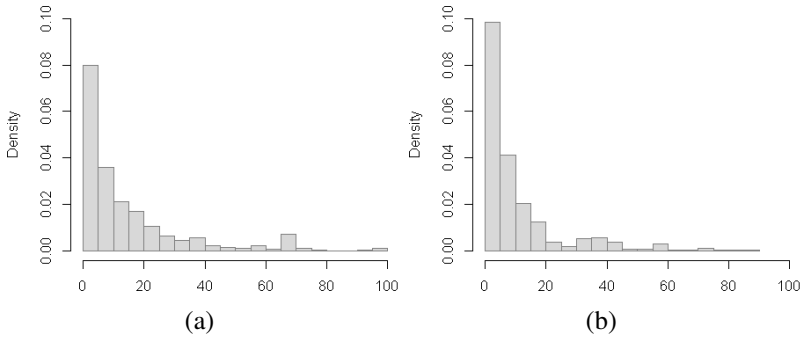


Figure 10: Histogram of the localization error using the best acquired ANN. (a) Using 10k generation in the PSO. (b) Using 100k generation in the PSO. The x axes represent the error in centimeters.

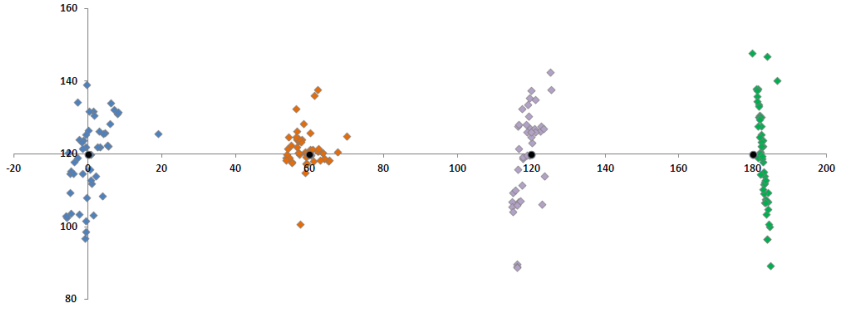


Figure 11: Section of the plane (Fig. 1(a)) with expected and obtained values for 4 coordinates using the best evolved ANN. The black dots are the expected value; diamonds show the obtained values. Each colour represents a different position. Axes x and y are in centimeters. For all positions in the full evaluated plane, 86% of the errors are below 20 cm.

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 (Pessin et al., 2011). 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 $\approx$ 45 cm to $\approx$ 15 cm. Also, we might see that the use PSO neighborhood topology and the non-uniform inertia 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 $\approx$ 10 cm.

Nevertheless, it is important to notice that results presented in this paper cannot be directly compared with results from (Pessin et al., 2011) 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

Deep, K., Arya, M., Bansal, J. C., 2011. A non-deterministic adaptive inertia weight in pso. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. GECCO '11. ACM, New York, NY, USA, pp. 1155–1162.

Dorofki, M., Elshafie, A. H., Jaafar, O., Karim, O. A., Mastura, S., 2012. Comparison of artificial neural network transfer functions abilities to simulate extreme runoff data. In: International Conference on Environment, Energy and Biotechnology.

Eberhart, R. C., Kennedy, J., Shi, Y., 2001. Swarm Intelligence. M. Kaufmann.

Elnahrawy, E., Li, X., Martin, R., 2004. The limits of localization using signal strength: a comparative study. In: IEEE SECON. pp. 406–414.

Engelbrecht, A. P., 2005. Fundamentals of Comp. Swarm Intelligence. Wiley.

Espinace, P., Soto, A., Torres-Torriti, M., 2008. Real-time robot localization in indoor environments using structural information. In: Robotic Symposium, 2008. LARS '08. IEEE Latin American. IEEE Computer Society, Los Alamitos, CA, USA, pp. 79–84.

Fogel, D., 2006. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press Series on Computational Intelligence. Wiley.

Fu, S., Hou, Z., Yang, G., 2009. An indoor navigation system for autonomous mobile robot using wsn. In: Networking, Sensing and Control. pp. 227–232.

Haykin, S. O., 2008. Neural Networks and Learning Machines. Prentice Hall.

Herrera, F., Zhang, J., 2009. Optimal control of batch processes using particle swam optimisation with stacked neural network models. Computers & Chemical Engineering 33 (10), 1593 – 1601.

Huang, G.-B., mar 2003. Learning capability and storage capacity of two-hidden-layer feedforward networks. Neural Networks, IEEE Transactions on 14 (2), 274 – 281.

Kennedy, J., 1997. The particle swarm: social adaptation of knowledge. In: Evolutionary Computation, 1997., IEEE International Conference on. pp. 303–308.

Kentzoglanakis, K., Poole, M., 2009. Particle swarm optimization with an oscillating inertia weight. In: Proceedings of the 11th Annual conference on Genetic and evolutionary computation. GECCO '09. ACM, New York, NY, USA, pp. 1749–1750.

Ladd, A., Bekris, K., Rudys, A., Wallach, D., Kavraki, L., 2004. On the feasibility of using wireless ethernet for indoor localization. Robotics and Automation, IEEE Trans. on 20 (3), 555 – 559.

Malik, R. F., Rahman, T. A., Hashim, S. Z. M., Ngah, R., 2007. New particle swarm optimizer with sigmoid increasing inertia weight. International Journal of Computer Science and Security 1, 35–44.

Martinelli, A., 2002. The odometry error of a mobile robot with a synchronous drive system. Robotics and Automation, IEEE Trans. on 18 (3), 399–405.

Mitchell, T. M., 1997. Machine Learning. McGraw-Hill.

Napier, A., Sibley, G., Newman, P., 2010. Real-time bounded-error pose estimation for road vehicles using vision. In: IEEE Conf. on Intelligent Transp. Systems.

Ozcan, E., Mohan, C., 1999. Particle swarm optimization: surfing the waves. In: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. Vol. 3.

Pant, M., Thangaraj, R., Abraham, A., 2009. Particle swarm optimization: Performance tuning and empirical analysis. In: Abraham, A., Hassanien, A.-E., Siarry, P., Engelbrecht, A. (Eds.), Foundations of Computational Intelligence Volume 3. Vol. 203 of Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 101–128.

Pessin, G., Osório, F. S., Souza, J. R., Costa, F. G., Ueyama, J., Wolf, D. F., Braun, T., Vargas, P. A., 2012. Evolving an indoor robotic localization system based on wireless networks. In: 13th International Conference on Engineering Applications of Neural Networks (EANN). pp. 1–10.

Pessin, G., Osório, F. S., Ueyama, J., Souza, J. R., Wolf, D. F., Braun, T., Vargas, P. A., 2011. 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.

Robles, J., Deicke, M., Lehnert, R., 2010. 3d fingerprint-based localization for wireless sensor networks. In: Positioning Navigation and Communication (WPNC).

Silveira, T., de Oliveira, H., Salgado, R., da Silva, L., Mateus, G., 2010. Cosine function applied to the inertia control in the particle swarm optimization. In: Evolutionary Computation (CEC), 2010 IEEE Congress on. pp. 1 –8.

Teoh, E., Tan, K., Xiang, C., nov. 2006. Estimating the number of hidden neurons in a feedforward network using the singular value decomposition. Neural Networks, IEEE Transactions on 17 (6), 1623 –1629.

Thrun, S., Fox, D., Burgard, W., Dellaert, F., 2001. Robust monte carlo localization for mobile robots. Artificial intelligence 128 (1-2), 99–141.

Yao, X., 1999. Evolving artificial neural networks. Proc. of IEEE 87 (9), 1423–1447.