Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

MAP MATCHING BASED ON PARACONSISTENT ARTIFICIAL NEURAL NETWORKS

Anderson Anjos da Silva [*] **Carlos Henrique Costa Ribeiro** [**]

[*] *NCROMA Research Group - Computer Science Division - Technological Institute of Aeronautics - ITA , Pça. Mal. Eduardo Gomes 50, Vila das Acácias 12228-900,São José dos Campos, SP, Brazil, (Tel: +55-012-39476887; e-mail: anjos@ita.br).*
[**] *(e-mail: carlos@ita.br)*

Abstract: This paper presents a new method for matching metric maps generated by mobile robots that act cooperatively. This process of information matching makes it possible to perform global map generation from local maps (possibly partial and nonconsistent) provided by individual robots. The proposed method is based on a paraconsistent artificial neural network model that considers as input data the Euclidean distances between the points from each one of the partial maps. The use of this kind of input information makes the individual maps invariant with respect to relative rotation and translation among the robots in the mapping environment. The neural network then analyzes these distances to determine what are the matching belief relations among the points of the distinct maps. The algorithm implemented for the neural architecture achieved good results with very satisfactory computational performance, and made it possible to determine the certainty and contradiction degress in the map point matching analysis. The results show that the proposed approach is robust for the cases were it was applied. Equally important is the fact that the considered architecture allows for the combination of information from partial maps acquired in execution time during navigation.

Keywords: Map Matching, Cooperative Mobile Robotics, Paraconsistent Logic, Artificial Neural Networks

1. INTRODUCTION

A fundamental issue to be considered for the design and implementation of autonomous mobile robots is the exploratory capacity in the actuation environment, with no prior information and aiming at learning a sufficient representation of the environment, as far as the task goals are concerned. In order to build up this representation, the robot collects and processes data informed by its onboard sensors. The final objective is to acquire, through this exploratory processes, a global map of the environent and the corresponding self-localization of the robot.

Many of the proposed solutions for map generation and localization are based on simultaneous algorithms (Simultaneous Localisation And Mapping — SLAM), e.g. Dissanayake et al. [2000] and Thrun and Liu [2003]. However, independently from considering either SLAM or separate mapping and localization techniques, it is a matter of fact that environmental modelling based on the combination of partial maps is expected to be computationally more efficient, due to some degree of parallelism, than modelling based on a single exploratory robot. Therefore, there is an iherent advantage on using coperative robotic systems where algorithms for matching partial individual maps are used to obtain a global envionment map, such as in the method described in Diosi and Kleeman [2005].

This article proposes a novel method for matching individual local maps generated by cooperative robots. The method is based on paraconsistent artificial neural networks, described in Section 3.

2. PARACONSISTENT LOGIC

According to da Costa et al. [1999] Batens et al. [2000] and Bremer [2005], Paraconsistent Logic is a non-classical Logic that was proposed to deal with realistic situations regarding uncertainties that are not supported by classical approaches.

Let T be a theory based on a logic L, from a language L′ that includes the negation symbol $\neg$. Theory T is said to be *nonconsistent* if there is a sentence A such that A and $\neg$A are theorems of T, otherwise T is said to be consistent.

We say that a theory T is trivial if all the sentences of its language are theorems, otherwise we say that T is nontrivial.

A logic L is paraconsistent if it is used as basis for nontrivial and nonconsistent theories, i.e., a paraconsistent logic allows for operations on inconsistent information systems without subsuming triviality of the theory.

2.1 Two-valued Annotated Paraconsistent Logic

A Two-valued Annotated Paraconsistent Logic is a paraconsistent logic with a representation on how much an evidence express knowledge about a proposition P, based on two components.

A proposition $P_{(\mu,\lambda)}$ is such that $\mu, \lambda \in [0, 1]$, μ indicates the degree of a favouring evidence for P and λ is the degree of oposing evidence for P. From those definitions, one can obtain:

- $P_{(1.0,0.0)}$, a true proposition (complete favouring evidence, no opposing evidence).
- $P_{(0.0,1.0)}$, a false proposition (no favouring evidence, complete opposing evidence).
- $P_{(1.0,1.0)}$, a nonconsistent proposition (complete favouring evidence, complete opposing evidence).
- $P_{(0.0,0.0)}$, a paracomplete proposition (no favouring or opposing evidences).
- $P_{(0.5,0.5)}$, a nondefined proposition (equal favouring and opposing evidences at 0.5).

Propositions in a two-valued paraconsistent logic can be depicted as points in the lattice shown in Figure 1.

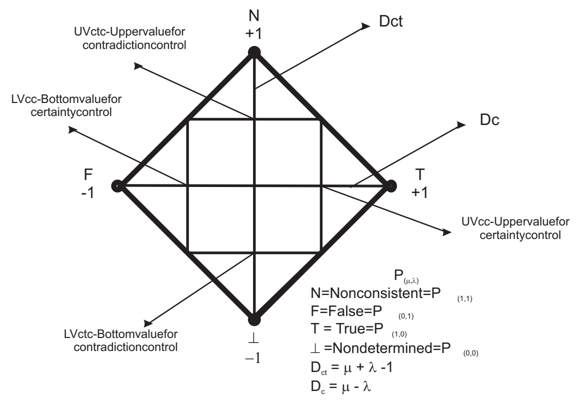
N
+1
Dct
UVctc-Uppervaluefor contradictioncontrol
LVcc-Bottomvaluefor certaintycontrol
Dc
F
-1
T
+1
UVcc-Uppervaluefor certaintycontrol
P_(μ,λ)
N=Nonconsistent=P (1,1)
F=False=P (0,1)
T = True=P (1,0)
⊥ =Nondetermined=P (0,0)
D_ct = μ + λ -1
D_c = μ - λ
LVctc-Bottomvaluefor contradictioncontrol
⊥
−1

Fig. 1. Lattice for interpreting propositions in a two-valued annotated paraconsistent logic.

In a two-valued paraconsistent logic, degrees of belief and disbelief are evidences which support the decision making process. Figure 1 also presents the equations for acquiring the degrees of contradiction D_{ct} and certainty D_c, respectively Equations 1 and 2.

$$D_{ct} = \mu + \lambda - 1 \tag{1}$$

$$D_c = \mu - \lambda \tag{2}$$

As shown on Figure 1, the values of the degree of certainty D_c are marked horizontally in the x-axis (degree of certainty axis), whereas the degree of contradiction are marked vertically (in the degree of contradiction axis). Two arbitrary limiting values (UV_{cc} = upper value for certainty control and LV_{cc} = lower value for certainty control) determine if the resulting degree of certainty is high enough to establish if the analysed proposition is absolutely true or false. Similarly, two limiting values (UV_{ctc} = upper value for contradiction control and LV_{ctc} = lower value for contradiction control) determine if the resulting degree of contradiction is high enough to establish if the analysed proposition is absolutely nonconsistent or nondetermined. Both degrees are on the interval [-1,+1], and an analysis on the point representation of any proposition in the lattice outputs the intensity of its degrees of certainty and contradiction

3. PARACONSISTENT ARTIFICIAL NEURAL NETWORKS

As defined in da Silva Filho and Abe [2001], Abe [2004], Abe et al. [2005] and Ferrara et al. [2005], a paraconsistent artificial neural network (PANN) is a connectionist structure, a large network of components based on Annotated Paraconsistent Logic (da Costa et al. [1999]). Such components are the paraconsistent artificial neural cells.

3.1 The Paraconsistent Artificial Neural Cell

The paraconsistent artificial neural cell (PANC) is the simplest structure in a PANN with a well-defined functionality da Silva Filho and Abe [2001]. The output value D'_c of a PANC is the resulting degree of belief, calculated from Equation 2 and limited to the range $[-1, +1]$. However, for obtaining the resulting degree of belief according to the two-valued paraconsistent logic formalism (Section 2.1), the value must be in the interval $[0, 1]$. A normalization is thus required, according to Equation 3.

$$D'_c = \frac{D_c + 1}{2} \tag{3}$$

Equation 3 is the Basic Structural Equation BSEq for a PANC.

Amongst the many PANC models described in da Silva Filho and Abe [2001], we present here only three types of PANC which are required for the design of a paraconsistent neural network for map matching: Simple Logical Connection Cell (cPANC), Decision Cell (dPANC) and the Learning Cell (lPANC).

3.2 The Paraconsistent Simple Logical Connection Cell

According to da Silva Filho and Abe [2001], the Paraconsistent Simple Logical Connection Cell (cPANC), represented in Figure 2, is a logical comparator among the degrees of belief that are input to it. For the particular case of two inputs (μ, λ), the cPANC generates the output according to:

```
If : $D'_c \geq 1/2$ then
    Output $\mu$
Else:
    Output is $\lambda$
```

3.3 The Paraconsistent Artificial Neural Cell for Decision

The Paraconsistent Artificial Neural Cell for Decision, depicted in Figure 3, receives as inputs two belief degrees (μ, λ) and outputs a result corresponding to a paraconsistent logical 3-valued decision. Value 1 is the conclusion "True", value 0 represents "False", and vale 1/2 represents "Nondefined". This cell is also equipped with two adjusting parameters: a decision factor Ft_d and a contradiction tolerance factor Ft_{ct}. The output is obtained in the following way:

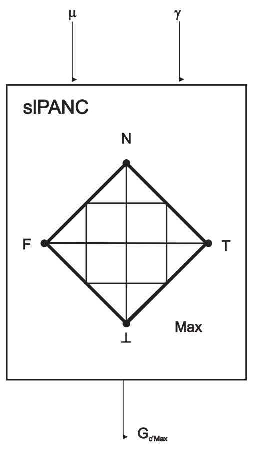

Fig. 2. The Paraconsistent Simple Logical Connection Cell.

- Calculation of limit values for falsehood and truth:

$$Vl_F = \frac{1 - Ft_d}{2} \tag{4}$$

$$Vl_T = \frac{1 + Ft_d}{2} \tag{5}$$

- Calculation of contradiction and certainty is performed according to equations 1 and 2.

The output states S_1 e S_2 are then obtained from the following comparisons:

$$If : Vl_F < D'_c < Vl_V \text{ then}$$

$$S_1 = 1/2 = \text{Nondefined and } S_2 = 0$$

$$If : D'_c \geq Vl_V \text{ then}$$

$$S_1 = 1 = \text{True and } S_2 = 0$$

$$If : D'_c \leq Vl_F \text{ then}$$

$$S_1 = 0 = \text{False and } S_2 = 0$$

$$If : |D_{ct}| \geq Ft_{ct} \text{ and } |D_{ct}| > |D_c| \text{ then}$$

$$S_1 = 1/2 = \text{ Nondefined and } S_2 = D_{ct} = |D_{ct}|$$

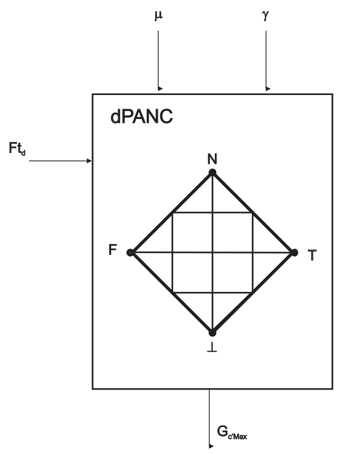

Fig. 3. Paraconsistent Artificial Neural Cell for Decision.

3.4 The Paraconsistent Artificial Neural Cell for Learning

A paraconsistent artificial neural cell for learning (lPANC) is a pattern learning structure. The method for determining its learning is based on the equations of Paraconsistent Logic. A lPANC is parameterized by an externally adjusted learning factor Fa.

A lPANC is an autoassociative memory for values in the closed interval [0,1]. Initially, its output is set at 0.5, indicating a nondefined output. As input values are fed to the cell, the output belief if forced convergence to the last input value.

For generating this autoassociation, the output belief is used in a feedback manner, with a complement calculation (C in Figure 4) in the disbelief input.

Figure 4 is the representation of a lPANC.

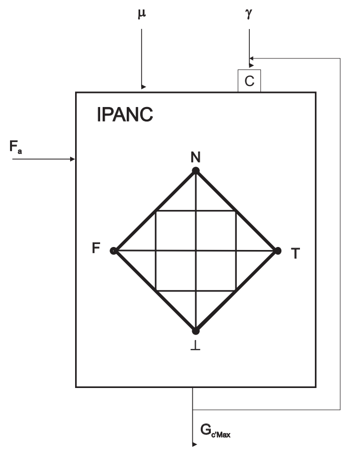

Fig. 4. Paraconsistent Artificial Neural Cell for Learning.

The learning process is based on Equation 6:

$$D'_c(k+1) = \frac{(\mu - (1 - D'_c(k)) * Fa + 1)}{2} \tag{6}$$

where $G'_c(k+1)$ is the value of the resulting belief, μ is the input pattern appied to the cell at a given time, and $1 - G'_c(k)$ is the negation of the previous resulting belief. ‘

3.5 Paraconsistent Artificial Neural Units

Paraconsistent Artificial Neural Units (PANUs) can be thought of as similar to the structural local arrangements of biological neurons in the nervous system. PANUs are clusters of PANCs purposefully linked, forming arrangements with distinct configurations and defined fuctions. Such units are then linked among themselves to form the basic functional structue of a PANN. In the work reported herein, we implemented three PANUs: one for decision making, one for learning and one for extraction of maxima.

4. MAP MATCHING

A global map of an explored environment can be obtained by combining local map information acquired through a cooperative robotic system da Silva et al. [2005]. Each individual robot explores the environment and generate corresponding local maps using standard techniques for map generation. Through cooperation one can obtain, from those partial and possibly inconsistent maps, a global map representation using an appropriate method for map matching.

4.1 Preprocessing

Map generation for each robot is based on the algorithm described in da Silva and Ribeiro [2007]. This algorithm is responsible for acquiring map reference points and distances between the robot and obstacles and walls in the environment via a laser scanner mounted on the top of the robot. From the x and y coordinates of the reference points obtained from the laser sensing, an array with all the pairwise point-to-point Euclidean distances is generated. Thus,

$$E(j,i) = \sqrt{(m(j,x) - m(i,x))^2 + (m(j,y) - m(i,y))^2} \qquad (7)$$

where i and j are the indexes for all the possible point pairs of the map.

These distances are the inputs for the PANN. Using distance information for map matching makes the process invariant to relative translation and rotation among the robots which are mapping the environment.

4.2 Modelling of the PANN

We propose a PANN for solving the map matching problem. The main characteristics of this approach are a) matching using as basis for the determination of certainty and contradiction degrees between maps in a pointwise manner; and b) low computational cost. Figure 5 illustrates the implemented PANN.

The model is based on three types of PANCs. The first one is a dPANC with adjustable decision factor which defines a threshold for the matching comparison between the point-to-point Euclidean distances. The outputs of the dPANCs are fed to lPANCs for a learning process which produces a convergence trend towards points with largest similarity. The last component is the cPANC, which defines the point with the largest degree of belief for matching the compared point.

The inputs for the PANN are two Euclidean distance arrays, each corresponding to a partial map generated by a robot. Lines and columns are indexed by the points i and j of the map, and each position in the array is the corresponding Euclidean distance from i to j, E(j,i).

The operation of the PANN is as follows.Initially, each line position of a line from the array of Euclidean distances from one map is compared against the line positions of each line from the array from the other map, via the dPANCs (which together form the decision PANU). For each comparison of distances that are below a threshold (decision factor), the resulting outputs are fedforward to the learning PANU formed by lPANCs, each of which forcing the output towards the point from the second map with largest similarity to the point from the first map among the analysed points corresponding to each line of the second array. Then, the resulting beliefs from the lPANCs are input to the connection PANU, composed by cPANCs, which then determine which maps points

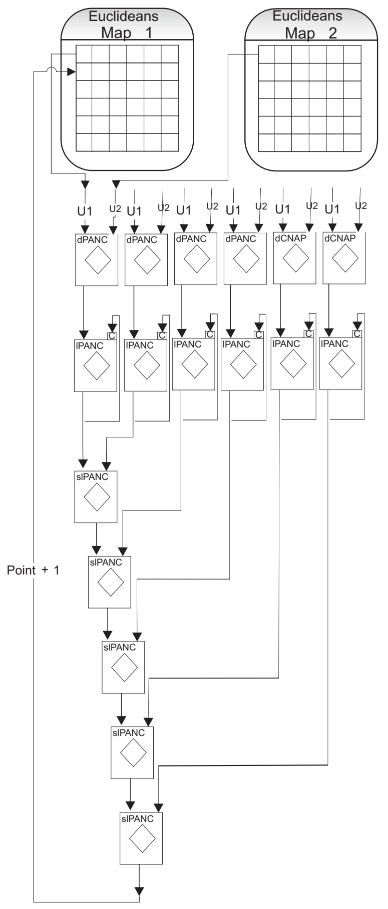

Fig. 5. Paraconsistent Artificial Neural Network for Point Matching.

from the analysed map has largest beliefs. This process is repeated for all the other points from the first map.

5. EXPERIMENTAL RESULTS

Experiments on map matching using the PANN were implemented in Matlab 7.0 (MathWorks [2007]). Sensor information was acquired by running the Player/Gazebo simulator Gerkey et al. [2004], using a model of a laser scanner on a Pioneer2DX robot which navigates around the simulated environment and periodically executes the PANN for matching the acquired points. Thus, although there is actually a single simulated robot, the matching process is carried out considering partial maps independently generated at distinct times, which is precisely what would happen if the local maps were generated by different robots.

Figures 6 and 7 show the local maps 1 and 2, generated at different times and robot locations. The decision factor for the dPANC was set as 0.03.

Executing the PANN generated the results informed in Table 1, which presents all the beliefs of points from Map 1 (in the rows) with respect to points in Map 2 (in the columns). The matching points are in bold, and the PANN identified the best possible matching.

After some navigation steps of the robot (again at different positions), we ran again the PANN. Representations of the

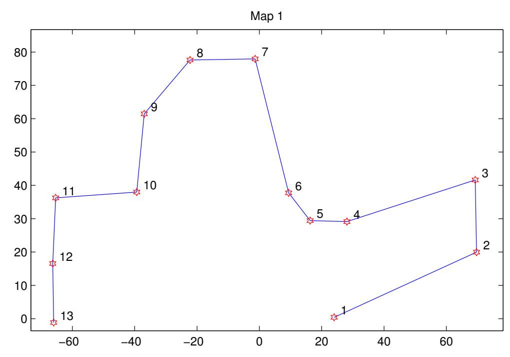

Fig. 6. Local Map 1.

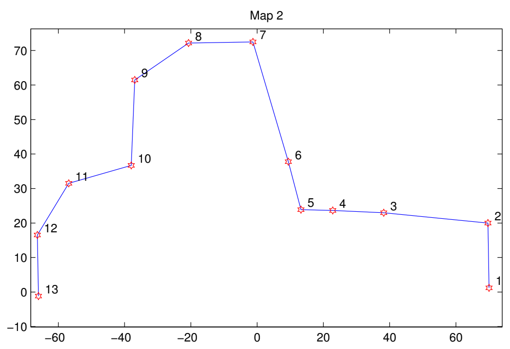

Fig. 7. Local Map 2.

Table 1. Beliefs obtained from the matching of Map 2 to Map 1.

Points	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0	0	0.4	0.9	0.9	0.9	0.9	0.8	0.8	0.9	0.5	0.3	0.6
2	**1**	0.9	0.9	0.5	0.5	0.6	0.6	0.9	0.9	0.9	0.9	0.8	0.9
3	0.9	**1**	0.9	0.8	0.6	0.6	0.9	0.7	0.9	0.8	0.5	0.8	0.8
4	0	0	**1**	0.9	0.4	0.7	0.9	0.8	0.8	0.9	0.7	0.8	0.8
5	0	0	0.2	**1**	0.9	0.8	0.3	0.9	0.8	0.8	0.6	0.9	0.8
6	0	0	0.1	0.6	**1**	**1**	0.4	0.9	0.9	0.8	0.6	0.8	0.8
7	0	0.1	0.7	0.9	0.8	0.9	**1**	0.9	0.9	0.9	0.9	0.8	0.9
8	0.1	0.1	0.9	0.9	0.8	0.9	0.9	**1**	0.9	0.5	0.9	0.9	0.9
9	0.1	0.1	0.9	0.5	0.6	0.8	0.7	0.9	**1**	0.9	0.9	0.9	0.9
10	0.1	0.1	0.8	0.1	0.5	0.3	0.2	0.5	0.9	**1**	0.9	0.9	0.9
11	0.8	0.8	0.3	0.9	0.9	0.9	0.9	0.8	0.9	0.8	**1**	0.9	0.9
12	0.8	0.8	0.4	0.9	0.9	0.8	0.9	0.9	0.9	0.9	0.9	**1**	0.9
13	0.8	0.8	0.6	0.9	0.8	0.8	0.9	0.9	0.9	0.9	0.9	0.8	**1**

generated local maps to be matched, Map 3 and Map 4, are shown in Figures 8 and 9.

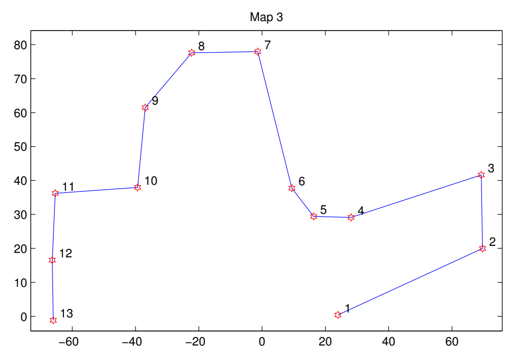

Fig. 8. Local Map 3.

Results obtained for the matching of maps 8 and 9 are shown in Table 2, which emphasizes the points from Map 4 with the best belief correspondences against the points in Map 3.

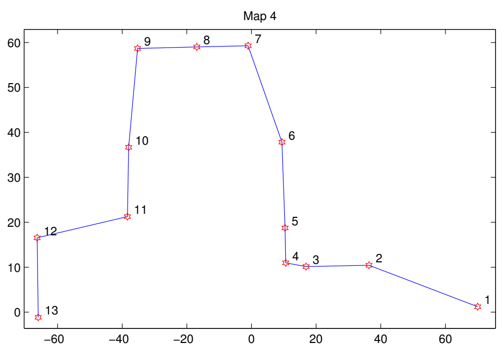

Fig. 9. Local Map 4.

Table 2. Beliefs obtained from the matching of Map 4 to Map 3.

Points	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0	0.5	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.3	0.6
2	0.8	0.5	0.6	0.2	0.2	0.7	0.7	0.3	0.9	0.9	0.9	0.8	0.9
3	**0.9**	0.9	0.6	0.6	0.7	0.7	0.9	0.8	0.9	0.8	0.5	0.8	0.8
4	0	**0.9**	0.2	0.9	0.9	0.5	0.6	0.8	0.8	0.8	0.8	0.8	0.8
5	0	0.2	**1**	**1**	0.9	0.9	0.9	0.8	0.8	0.8	0.6	0.8	0.8
6	0	0.2	0.9	0.9	**1**	**1**	0.8	0.9	0.8	0.8	0.6	0.8	0.8
7	0.1	0.6	0.9	0.6	0.6	0.9	**1**	0.9	0.9	0.9	0.9	0.8	0.9
8	0.1	0.9	0.9	0.6	0.6	0.9	0.9	**1**	0.9	0.5	0.9	0.9	0.9
9	0.2	0.6	0.6	0.9	0.9	0.9	0.9	0.9	**1**	**1**	0.9	0.9	0.9
10	0.2	0.9	0.4	0.7	0.5	0.4	0.5	0.9	0.9	**1**	**1**	0.9	0.9
11	0.8	0.7	0.9	0.9	0.8	0.8	0.8	0.9	0.9	0.8	0.9	**1**	0.9
12	0.8	0.6	0.9	0.8	0.8	0.9	0.9	0.6	0.9	0.9	0.9	0.9	**1**
13	0.8	0.7	0.8	0.5	0.8	0.9	0.9	0.6	0.9	0.9	0.6	0.8	0.9

Again, with more exploration of the environment and further approximation to the obstacles, new points for matching were acquired. The new maps (respectively Map 5 and Map 6) are shown in Figures 10 and 11.

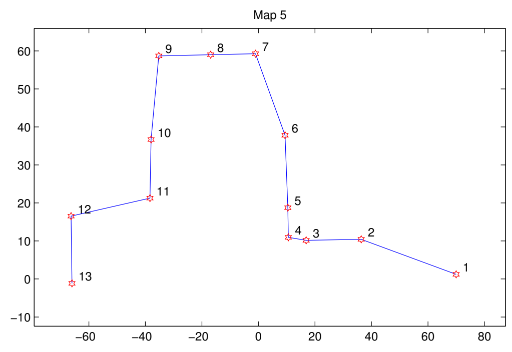

Fig. 10. Local Map 5.

According to the results depicted in Table 3, we can again say that the best matching was generated, through a correct pointwise belief identification.

Finally, it is worth pointing out that the PANN model had very good computational performance, which might encourage real-time map matching applications. Results were produced on a computer with Intel Core Duo 1.6 and 1Gb RAM, with proessing times of approximately 0.5 seconds.

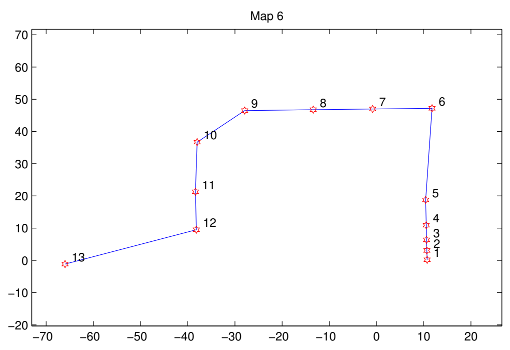

Fig. 11. Local Map 6.

Table 3. Beliefs obtained from the matching of Map 6 to Map 5.

Points	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**
1	0.6	0.6	0.6	0.6	0.3	0.6	0.3	0.5	0.5	0.5	0.5	0.8	0.8
2	0.6	0.6	0.6	0.6	0.5	0.6	0.6	0.7	0.9	0.4	0.6	0.8	0.8
3	0.8	0.6	0.8	0.8	0.8	0.4	0.8	0.6	0.7	0.9	0.4	0.6	0.6
4	**1**	0.9	0.9	0.9	0.8	0.4	0.8	0.8	0.7	0.8	0.4	0.8	0.8
5	0.9	**0.9**	**1**	**1**	0.8	0.4	0.8	0.8	0.6	0.7	0.6	0.7	0.7
6	0.9	0.8	0.9	0.9	**0.9**	0.7	0.8	0.8	0.4	0.9	0.7	0.8	0.9
7	0.8	0.8	0.8	0.9	0.8	**1**	**0.9**	0.4	0.5	0.8	0.9	0.9	0.9
8	0.6	0.6	0.6	0.7	0.7	0.5	0.6	**0.9**	**1**	0.9	0.9	0.6	0.7
9	0.6	0.5	0.5	0.6	0.3	0.6	0.5	0.7	0.6	0.7	0.9	0.6	0.7
10	0.6	0.6	0.7	0.7	0.7	0.4	0.7	0.4	0.5	**1**	**1**	0.6	0.7
11	0.9	0.7	0.7	0.7	0.7	0.4	0.8	0.7	0.5	0.7	0.9	**0.9**	0.9
12	0.6	0.6	0.6	0.6	0.6	0.4	0.7	0.7	0.5	0.6	0.7	0.9	**1**
13	0.6	0.6	0.6	0.6	0.6	0.6	0.7	0.9	0.6	0.7	0.9	0.5	0.7

6. CONCLUSIONS AND FUTURE WORK

We proposed a new map matching algorithm based on paraconsistent logic and information acquired by cooperative robots exploring an unknown environment.

From the individual maps generated by each robot, we initially generate point-to-point Euclidean distances. This corresponds to the production of a set of information arrays which are then fed to a paraconsistent artificial neural network whose aim is to produce the map matching. From the distance information in each array, this network determines the pairwise degrees of similarity among points in the maps. As a net result, the degrees of matching belief among the points of the maps are generated.

We performed three experiments with data acquired from a laser scanner on a Pioneer2DX robot, simulated in the Player/Gazebo platform, with good results. Optimal matching was consistently produced, with low computational cost.

For future work, we intend to consider other input representation techniques distinct from Euclidean distance. We also intend to consider learning, namely how to adapt the learning cell in order to produce more accurate matchings.

7. ACKNOWLEDGEMENTS

The authors are grateful to CAPES and FAPESP for the financial support to carry out this research.

REFERENCES

Jair Minoro Abe. Paraconsistent artificial neural networks: An introduction. In *KES*, pages 942–948, 2004.

Jair Minoro Abe, Neli Regina Siqueira Ortega, Maurício C. Mário, and Marinho Del Santo. Paraconsistent artificial neural network: An application in cephalometric analysis. In Rajiv Khosla, Robert J. Howlett, and Lakhmi C. Jain, editors, *KES (2)*, volume 3682 of *Lecture Notes in Computer Science*, pages 716–723. Springer, 2005.

Diderik Batens, Chris Mortensen, Graham Priest, and Jean-Paul Van Bendegem. *Frontiers of Paraconsistent Logic*. Research Studies Press, London, 2000.

Manuel Bremer. *An Introduction To Paraconsistent Logics*. Peter Lang GmbH, Frankfurt - Germany, 2005.

Newton C. A. da Costa, Jair Minoro Abe, João Inácio da Silva Filho, Afrânio Carlos Murolo, and Casemiro Fernando S. Leite. *Lógica Paraconsistente Aplicada*. Atlas, Brasil, 1 edition, 1999.

Anderson Anjos da Silva, Esther Luna Colombini, and Carlos Henrique Costa Ribeiro. Cognitive map merging for multi-robot navigation. *1st International Workshop on Multi-Agent Robotic Systems (MARS 2005) - 2nd International Converence on Informatics in Control, Automation and Robotics (ICINCO), 2005, Barcelona. Proceedings of the 1st International Workshop on Multi-Agent Robotics Systems (MARS 2005)*, pages 102–111, 2005.

Anderson Anjos da Silva and Carlos Henrique Costa Ribeiro. Mapeamento e localização simultâneos baseados em combinações de leituras polares. *VIII Simpósio Brasileiro de Automação Inteligente - SBAI, 2007. Anais do VIII Simpósio Brasileiro de Automação Inteligente*, 2007.

João Inácio da Silva Filho and Jair Minoro Abe. *Fundamentos das Redes Neurais Artificiais Paraconsistentes*. Arte e Ciência, Brasil, 1 edition, 2001.

Albert Diosi and Lindsay Kleeman. Laser scan matching in polar coordinates with application to slam. *Intelligent Robots and Systems, 2005. (IROS 2005). 2005*, IEEE/RSJ International Conference on:3317 – 3322, Aug 2005.

Gamini Dissanayake, Hugh Durrant-Whyte, and Tim Bailey. A computationally efficient solution to the simultaneous localisation and map building (slam) problem. *International Conference on robotics and automation*, 2000.

Luís Fernando Pompeo Ferrara, Keiji Yamanaka, and João Inácio da Silva Filho. A system of recognition of characters based on paraconsistent artificial neural networks. *Advances in Logic Based Intelligent Systems, 2005 Select Papers of LAPTEC 2005 - Frontiers in Artificial Intelligence and Its Applications, ISSN 0922-6389 IOS Press*, pages 127–134, 2005.

Brian Gerkey, Andrew Howard, Richard Vaughan, Nate Koenig, and Andrew Howard. Player/stage project, 2004. Disponível em: <http://playerstage.sourceforge.net/>. Acesso em: Fevereiro 2005.

MathWorks. *Getting Started With Matlab 7*. The MathWorks, Inc, 2007.

S. Thrun and Y. Liu. Multi-robot SLAM with sparse extended information filers. In *Proceedings of the 11th International Symposium of Robotics Research (ISRR'03)*, Sienna, Italy, 2003. Springer.