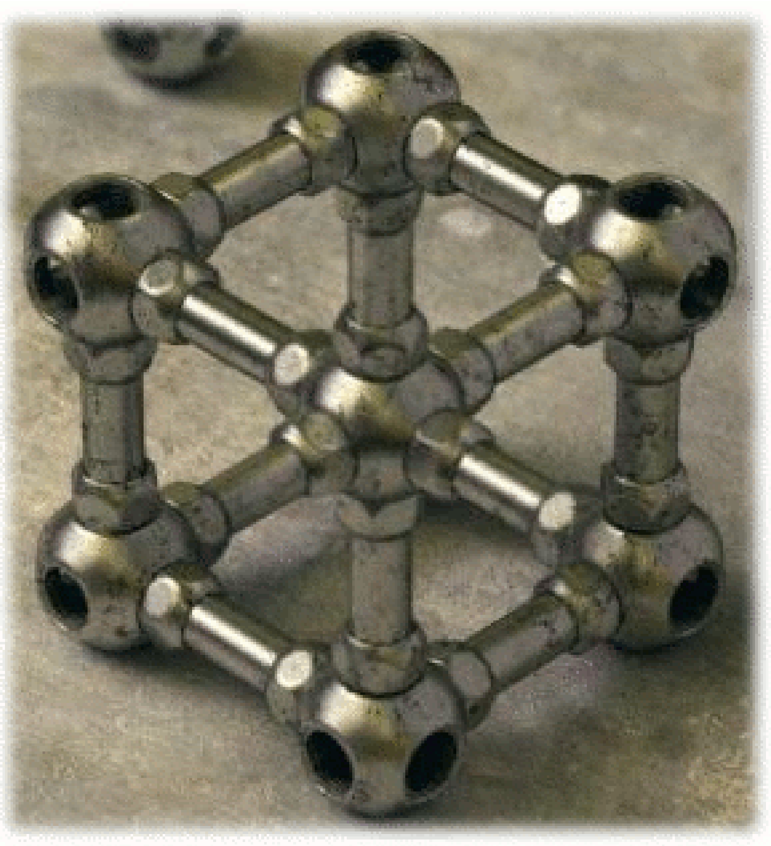


Combining Conflicting and Confirmatory Information for Map Matching Using Paraconsistent Neural Networks [1]

Author(s):

Anderson A. Silva
Anna H. R. Costa
Carlos H. C. Ribeiro

[1]This work was supported by Fapesp Project LogProb, grant 2008/03995-5, São Paulo, Brazil.

Combining Conflicting and Confirmatory Information for Map Matching Using Paraconsistent Neural Networks

Anderson A. Silva [*] **Anna H. R. Costa** [**]
Carlos H. C. Ribeiro [***]

[*] *Artificial Intelligence and Robotics Group - Technological Institute of Aeronautics, Pca. Mal. Eduardo Gomes 50, 12228-900,Sao Jose dos Campos, Brazil, (Tel: +55 12 39476887; e-mail: anjos@ita.br)*
[**] *Polytechnic School, University of Sao Paulo, Sao Paulo, Brazil, (Tel: +55 11 30915387; e-mail: anna.reali@poli.usp.br).*
[***] *Artificial Intelligence and Robotics Group - Technological Institute of Aeronautics, Pca. Mal. Eduardo Gomes 50, 12228-900,Sao Jose dos Campos, Brazil, (Tel: +55 12 39475895; 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 method is based on a paraconsistent artificial neural network model that considers as input preprocessed information from measurement data on landmark distances, possibly generated by different sensors in different robots and considering different metrics. The neural network then analyzes these inputs to determine what are the matching belief and contradiction relations among the points of the distinct maps. The algorithm implemented for the neural architecture achieved good results with satisfactory computational performance in the reported experiments, that consider combination of information from different linear distance metrics (Euclidean and Manhattan) and angle measurements. As a side effect, it made it possible to determine certainty and contradiction degrees for each map point match analysis, a feature that can be useful for decision making. 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

An important aspect to be considered for the implementation of autonomous mobile robots is the exploratory capacity in an unknown operation environment, aiming at learning a sufficient representation for it, as far as the task goals are concerned. In order to build up this representation, the robots collect and process data informed by their onboard sensors. The final objective is to acquire, through this exploratory processes, a global map of the environment and the corresponding self-localization of the robots.

Many of the proposed solutions for map generation and localization are based on algorithms that simultaneously build up the map and produce localization, in a bootstrap manner (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 inherent advantage on using cooperative robotic systems where algorithms for matching partial individual maps are used to obtain a global environment 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 a paraconsistent artificial neural network architecture, described in Section 3. Fundamentally, it considers as input preprocessed information from measurement data on landmark distances, possibly generated by different sensors in different robots and based on different metrics. The neural network then analyzes these inputs to determine what are the matching belief and contradiction relations among the points of the distinct maps.

The rest of this paper is organized as follows. Section 2 introduces the basic principles of paraconsistent logic and presents a two-valued paraconsistent logic model that is the basic theoretical underpinning for the map matching approach presented herein. Section 3 presents the paraconsistent artificial neural network architecture through its components (cells) and the corresponding algorithms.

2. PARACONSISTENT LOGIC

According to Batens et al. (2000) and Bremer (2005), Paraconsistent Logic is a non-classical logic 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 ¬. Theory T is said to be *nonconsistent* if there is a sentence A such that A and ¬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. Finally, 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 — based on two components — on how much an evidence express knowledge about a proposition $P_{(\mu,\lambda)}$. Here, $\mu, \lambda \in [0, 1]$, μ indicates the degree of favouring evidence for P and λ is the degree of opposing 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 (neither favouring or opposing evidences).
- $P_{(0.5,0.5)}$, a nondefined proposition (equal favouring and opposing evidences at 0.5).

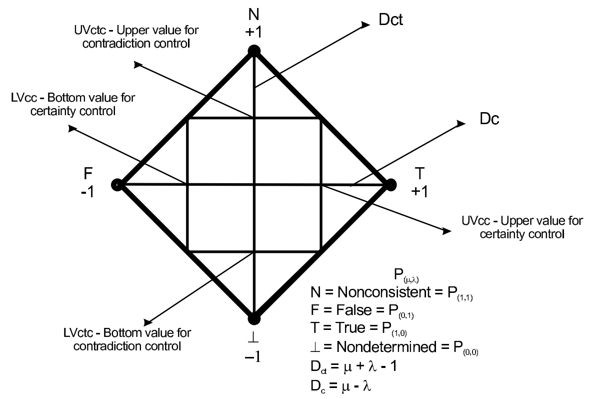
Propositions in a two-valued paraconsistent logic can be depicted as points in the lattice shown in Figure 1. In a two-valued paraconsistent logic, degrees of belief and disbelief are evidences which support the decision making process. Degrees of contradiction D_{ct} and certainty D_c are given respectively by

$$D_{ct} = \mu + \lambda - 1 \qquad (1)$$

and

$$D_c = \mu - \lambda \qquad (2)$$

As shown on Figure 1, the values of the degree of certainty D_c is marked horizontally in the x-axis (degree of certainty axis), whereas the degree of contradiction is 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 analyzed 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 analyzed 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.



N
+1
Dct
UVctc - Upper value for contradiction control
LVcc - Bottom value for certainty control
Dc
F
-1
T
+1
UVcc - Upper value for certainty control
LVctc - Bottom value for contradiction control
⊥
−1
$P_{(\mu,\lambda)}$
N = Nonconsistent = $P_{(1,1)}$
F = False = $P_{(0,1)}$
T = True = $P_{(1,0)}$
⊥ = Nondetermined = $P_{(0,0)}$
$D_{ct} = \mu + \lambda - 1$
$D_c = \mu - \lambda$

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

3. PARACONSISTENT ARTIFICIAL NEURAL NETWORKS

As defined in Abe (2004) and Abe et al. (2005), a paraconsistent artificial neural network (PANN) is a network of components based on annotated paraconsistent logic. 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. 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 $[0, 1]$, and thus a normalization is required:

$$D'_c = \frac{D_c + 1}{2} \qquad (3)$$

Equation 3 is the basic structural equation (BSEq) for a PANC.

We present here five types of PANC which are required for the design of a paraconsistent neural network for map matching and contradiction determination, namely: Simple Logical Connection Cell for Maximization (slPANCmax), Simple Logical Connection Cell for Minimization (slPANCmin), Decision Cell (dPANC), Analytic Connection Cell (acPANC) and Learning Cell (lPANC) .

3.2 The Simple Logical Connection Cell for Maximization

The Paraconsistent Simple Logical Connection Cell for Maximization (slPANCmax), 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 slPANCmax generates the output according to:

$$If\ D'_c \geq 1/2\ then\ output\ \mu\ else\ output\ \lambda \tag{4}$$

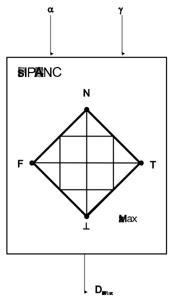


Fig. 2. The Paraconsistent Simple Logical Connection Cell for Maximization.

3.3 The Simple Logical Connection Cell for Minimization

The Paraconsistent Simple Logical Connection Cell for Minimization (slPANCmin), represented in Figure 3, is a logical comparator among the degrees of belief that are input to it. For the particular case of two inputs (μ, λ), the slPANCmin generates the output according to:

$$If\ D'_c \leq 1/2\ then\ output\ \mu\ else\ output\ \lambda \tag{5}$$

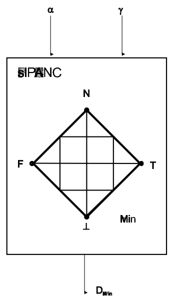


Fig. 3. The Paraconsistent Simple Logical Connection Cell for Minimization.

3.4 The Decision Cell

The Paraconsistent Artificial Neural Cell for Decision, depicted in Figure 4, receives as inputs two belief degrees (μ, λ) and outputs a result corresponding to a paraconsistent logical 3-valued decision.

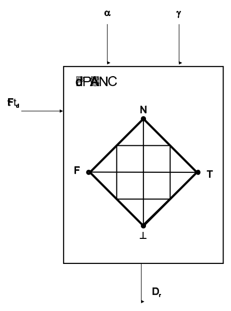


Fig. 4. Paraconsistent Artificial Neural Cell for Decision.

Value 1 is the conclusion "True", value 0 represents "False", and value 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}. For obtaining the output, we first calculate the limit values for falsehood $Vl_F = \frac{1-Ft_d}{2}$ and truth $Vl_T = \frac{1+Ft_d}{2}$ and then calculate contradiction and certainty according to equations 1 and 2. The output states S_1 e S_2 are then obtained from the following comparisons:

$$\text{If } Vl_F < D'_c < Vl_V \text{ then } S_1 = 1/2,\ S_2 = 0$$

$$\text{If } D'_c \geq Vl_V \text{ then } S_1 = 1,\ S_2 = 0$$

$$\text{If } D'_c \leq Vl_F \text{ then } S_1 = 0,\ S_2 = 0$$

$$\text{If } |D_{ct}| \geq Ft_{ct} \text{ and } |D_{ct}| > |D_c| \text{ then } S_1 = 1/2,\ S_2 = |D_{ct}|$$

3.5 The Analytic Connection Cell

The Paraconsistent Analytic Connection Cell acPANC, depicted in Figure 5 has a role in making the interconnection between cells of the neural network, involving degrees of belief as the objectives of the analysis. Each cell examines two values of degrees of belief which are applied at the input.

This cell has two tolerance factor inputs: Ft_c (certainty tolerance factor)and Ft_{ct} (contradiction tolerance factor), and produces its output according to:

$$\text{If } V_{icc} \leq D'_c < V_{scc} \text{ then } S_1 = D_c,\ S_2 = 0$$

$$\text{If } |D_{rct}| \leq Ft_{ct} \text{ and } |D_{rct}| > |D_c| \text{ then } S_1 = 1/2,\ S_2 = |D_c|$$

$$\text{else } S_1 = 1/2,\ S_2 = 0$$

where $V_{icc} = \frac{1+Ft_c}{2}$ and $V_{scc} = \frac{1+Ft_{ct}}{2}$.

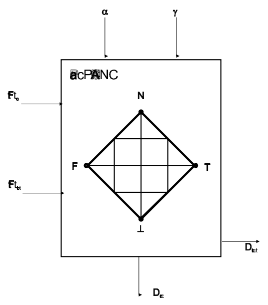


Fig. 5. The Paraconsistent Analytic Connection Cell.

The result is a single belief degree obtained only with BSEq. This unique output value, in turn, is a new degree of belief that will be used in other cells. The connection cell is therefore the link that allows different PANNs to process information in a distributed fashion.

3.6 The Learning Cell

The Paraconsistent Artificial Neural Cell for Learning (lPANC) (see Figure 6) is a pattern learning structure parameterized by an externally adjusted learning factor Ft_l. 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. For generating this autoassociation, the output belief is used in a feedback

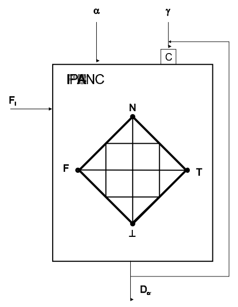


Fig. 6. The Learning Cell.

manner, with a complement calculation (C in Figure 6) in the disbelief input.

The learning process is based on Equation 6:

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

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

3.7 Paraconsistent Artificial Neural Units

Paraconsistent Artificial Neural Units (PANUs) are clusters of PANCs purposefully linked, forming arrangements with distinct configurations and defined functions. Such units are then linked among themselves to form the basic functional structure of a PANN. In the work reported herein, we implemented four PANUs: one for decision making, one for learning, one for extraction of maxima and one for contradiction determination.

4. MAP MATCHING

A global map of an explored environment can be obtained by combining local maps 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 Arleo et al. (1999). This algorithm is responsible for acquiring map reference points and distances between the robot and obstacles and walls in the environment. For the experiments reported herein, distance calculation is performed by 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, arrays with all the pairwise point-to-point distances can be generated. For the sake of experimental validation of the map matching technique, we consider here Euclidean and Manhattan distances and angles between landmarks.

For the Euclidean distances, we have:

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

For the Manhattan Distance array between the map points,we calculate:

$$M(j,i) = \sum |(m(j,x) - m(i,x))| + |(m(j,y) - m(i,y))|) \tag{8}$$

Finaly, the angle defined by the map points is:

$$M(j,i) = atan(abs(m(j,x) - m(i,x))), |(m(j,y) - m(i,y))|) \tag{9}$$

Indexes i and j are defined for all the possible point pairs of the map.

4.2 Modelling the PANN for Point Matching

We propose a PANN for solving the map matching problem which is based on the pointwise (landmark-based) determination of certainty and contradiction degrees between maps. Figure 7 illustrates the implemented PANN architecture.

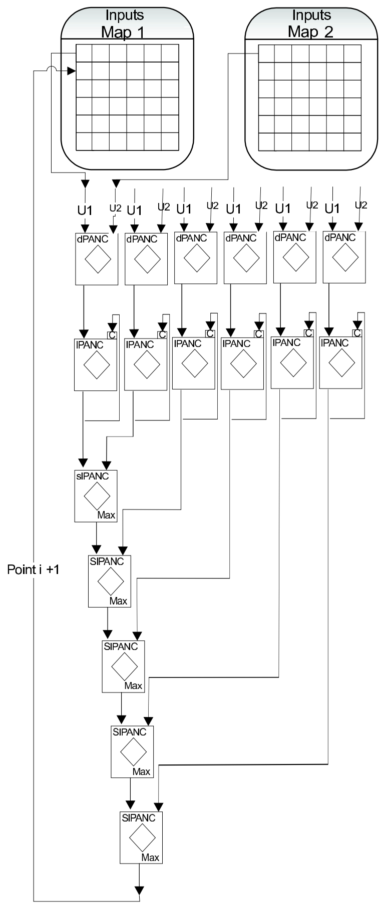


Fig. 7. PANN for Pointwise Matching.

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 among the points of the input array. 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 the arrays described in 4.1, each array 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 distance or angle values from i to j.

The operation of the PANN is as follows. Initially, each line position of a line from the array from one map is compared against the line positions of each line from the array from another 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 analyzed 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 map points from the analyzed map has largest beliefs. This process is repeated for all the other points from the first map.

4.3 Modelling the PANN for Contradiction Determination

The best matches from each metric used for map comparison (in our case Euclidean distance, Manhattan distance and angle) are inserted into another PANN (Figure 8), which determines if there are contradictions among the results. Notice that the choice of those metrics was arbitrary and for the sake of illustration. In general, different map matches might correspond, for instance, to different sensing techniques from different robots.

The first stage of the network determines the input pair with the lowest degree of contradiction. It is composed by acPANC and slPANC cells. The second stage produces the final output. From the inputs to the first stage that produced the lowest degree of contradiction, the second stage (using slPANC cells) determines the pair that produced the maximum degree of belief. Thus, the final output is the matching that produced the maximum degree of belief among those with the lowest degree of contradiction.

5. EXPERIMENTAL RESULTS

Experiments were implemented in Matlab 7.0 (MathWorks (2007)), with sensor information 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 running the PANN for matching the acquired points. 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 9 and 10 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.

Figures 11, 12 and 13, show the map mergings obtained from the PANN for point matching. Then, running the PANN for contradiction determination generated the results in Table 1, which presents all the point matchings between Map 1 and Map 2 after the contradiction analysis.

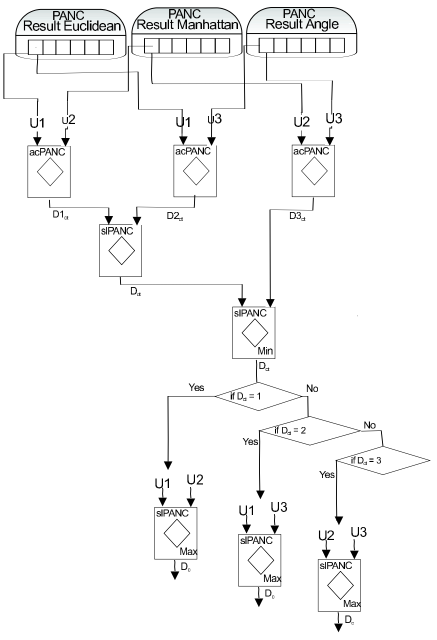


Fig. 8. PANN for Contradiction Determination.

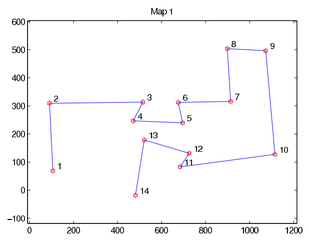


Fig. 9. Local map 1.

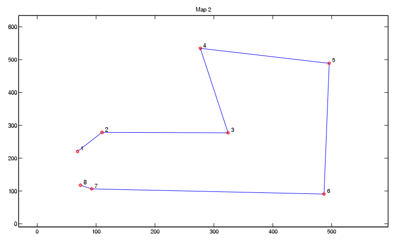


Fig. 10. Local map 2.

It can be noticed that the matching was produced with high belief degrees. Figure 14 shows the final matching produced after the contradiction analysis.

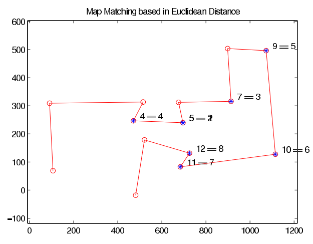


Fig. 11. Matching results for Euclidean distances.

Finally, it is worth pointing out that the PANN model had very good computational performance, which might

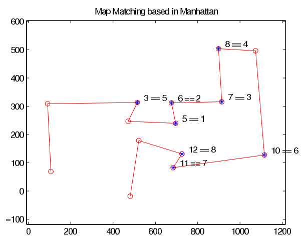


Fig. 12. Matching results for Manhattan distances.

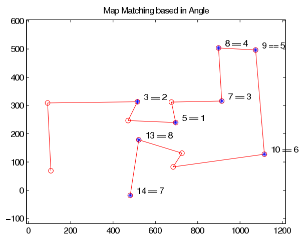


Fig. 13. Matching results for angle measurements.

Table 1. Results for point matching with contradiction analysis. The first and second columns are the indexes of the matched point. The third column shows the associated belief degrees.

Map 1	**Map 2**	D_c
1	5	0.9961
2	6	0.9961
3	7	0.9648
4	8	0.8750
5	9	0.8984
6	10	0.9961
7	11	0.9961
8	12	0.9023

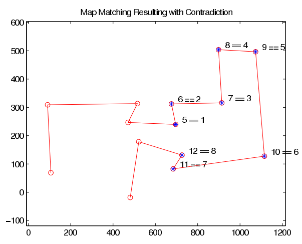


Fig. 14. Map matching after the contradiction analysis.

encourage real-time map matching applications. Results were produced on a computer with Intel Core Duo 1.6 and 1Gb RAM, with processing times of approximately 0.5 seconds.

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 the point-to-point Euclidean distances, Manhattan distances and angles. 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 angle and 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. This results are inserted in the contradiction PANN to analysis, for further disambiguation. We performed experiments with data acquired from a laser scanner on a Pioneer2DX robot, simulated in the Player/Gazebo platform, with adequate matching consistently produced with low computational cost.

For future work, we intend to extend the tests on the PANN architecture for multiple robots using different sensing capabilities. We also intend to develop a complexity analysis of the algorithm in order to formally assess its adequacy for real time map matching.

ACKNOWLEDGEMENTS

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

REFERENCES

Abe, J.M. (2004). *Paraconsistent Artificial Neural Networks: An Introduction*, volume 3214 of *Lecture Notes in Computer Science*. Springer Berlin.

Abe, J.M., Ortega, N.R.S., Mario, M.C., and Santo, M.D. (2005). Paraconsistent artificial neural network: An application in cephalometric analysis. In R. Khosla, R.J. Howlett, and L.C. Jain (eds.), *KES Conference*, volume 3682 of *Lecture Notes in Computer Science*, 716–723. Springer.

Arleo, A., del R. Millan, J., and Floreano, D. (1999). Efficient learning of variable-resolution cognitive maps for autonomous indoor navigation. *IEEE Transactions on Robotics and Automation*, 15(6), 990–1000.

Batens, D., Mortensen, C., Priest, G., and Bendegem, J.P.V. (2000). *Frontiers of Paraconsistent Logic*. Research Studies Press, London.

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

da Silva, A.A., Colombini, E.L., and Ribeiro, C.H.C. (2005). Cognitive map merging for multi-robot navigation. In *Procs. of the 1st Int. Workshop on Multi-Agent Robotics Systems (MARS 2005)*, 102–111. Barcelona.

Diosi, A. and Kleeman, L. (2005). Laser scan matching in polar coordinates with application to slam. In *Procs. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2005)*, 3317 – 3322. Australia.

Dissanayake, G., Durrant-Whyte, H., and Bailey, T. (2000). A computationally efficient solution to the simultaneous localisation and map building (slam) problem. *Int. Conf. on Robotics and Automation.*

Gerkey, B., Howard, A., Vaughan, R., Koenig, N., and Howard, A. (2004). The player/stage project. Available at: <http://playerstage.sourceforge.net/>. Access in: February 2005.

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

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