RESCUE
Cooperative Navigation for Rescue Robots
3rd Year Technical Report

A. Bernardino, L. Custódio, J. Frazão,
P. Lima, M. I. Ribeiro,
J. Santos-Victor, A. Vale

December 12, 2003

1 Introduction

During its third year, the research work on the RESCUE project was spread over several topics of short and long term application to real robots and scenarios, some of them new within the project, others extending past activities. This slightly more detailed report of the activity carried out during year 3 summarizes the progresses made. It is composed of three main sections (navigation, distributed planning and architectures) and two appendices with the full version of technical reports describing the work on the vision-based navigation of the aerial robot and the software plus functional architectures. The latter is the final activity report of the project grantee, João Frazão, whose grant within the project finished last October 2003.

In Section 2, the progresses on both the vision-based navigation of the aerial (blimp) robot and the topological navigation and mapping of the land robot are presented. Section 3 introduces work on distributed planning for a multi-robot rescue team, based on distributed Artificial Intelligence and Multi-Agent Systems techniques, and tested on a simulator which emulates scenarios that could well be the subject of future applications using real robots, as an extension of the current project. Finally, Section 4 summarizes the concepts concerning the software and functional architectures that will support the project development, and its implementation.

2 Navigation

2.1 Vision-Based Control of the Aerial Robot

To perform positioning or trajectory following tasks with an aerial robot, the development of control algorithms that overcome the underlying limitations of the system dynamics and kinematics, as well as the external disturbances, is required. In the RESCUE project, the only sensor used is a vision system consisting of a micro-camera placed onboard the aerial blimp robot, whose images are subject to real-time processing. From the homographies between consecutive images and assuming some prior information regarding the surrounding environment, it is possible to estimate velocities and displacements of the robot in 3D space.

Control methodologies were developed for the aerial robot that enable the system to accomplish positioning or trajectory following tasks, surpassing some limitations imposed by the physical system and the sensor. Image processing algorithms that enable obtaining the vehicle pose (position + orientation) and velocity were studied. Several types of linear and non-linear controllers were used to control the vehicle velocity, as well as its heading in 3D space. Two strategies for the reference definition were proposed, one based in position and coordinates in Cartesian space and the other based in image measurements, thus avoiding the need for high precision camera calibration.

The work developed consisted of system modelling and parameter identification, as well as control and image processing tests in a special-purpose simulator developed within the project. Furthermore, experiments were made with the real setup in which the algorithms were implemented, running in real-time.

More detailed information on this work can be found in Appendix A.

2.2 Land Robot Localization and Mapping

To support the navigation of the land robot a topological map was developed based on a probabilistic approach on the world representation as described in [2]. The robot perception is condensed in observations, o_t, a vector where each component relates to a different feature. This perception has to be recorded in a map, that is composed by a set of states. The robot estimated location is the map's state that most likely produced the observations acquired by the robot sensors during a given time interval. The proposed localization procedure, detailed in [4], was developed in previous years of the project.

In the project third year, the work concentrated on the mapping aspects that support the navigation, in particular those of feature extraction and feature selection. These issues are described in the sequel.

2.2.1 Feature Extraction

Topological maps provide useful abstractions of an environment, showing natural features that characterize particular locations or places. The algorithm developed in the frame of the RESCUE project, [1], is intended to adapt to the available sensors, this meaning that adding or removing different types of sensors enlarges or reduces the number of properties available to the algorithm. The raw data provided by the sensors requires a signal processing procedure before the implementation of the map building.

The projection of high-dimensional data onto low dimensional subspaces is called feature extraction. This extraction causes a loss of information in most cases. A feature extraction method must have the following properties [5]:

- Robustness to small displacements of the robots by means of capturing relevant features of the environment,
- Invariant to rotation,
- Invariant to lighting conditions,

- Invariant to occlusion,
- Fast computation,
- Capacity to compress the images as much as possible while retaining pertinent information.

Several features can be used on topological navigation: geometric features such as lines, corners, edges, shapes and other kind of features as color, textures and whatever can be considered a feature to discriminate a landmark. In outdoor environments, we should take into account which are good features to represent a landmark and carry out a selection procedure. These features can be different from those used in indoor environments.

An important goal for mapping is to achieve a good and optimized representation of features to improve the performance of matching processes and store the feature data in a small space. The feature vector f_t, is extracted at each time instant t, from the observation data o_t, by a nonlinear function FE:

$$f_t = FE(o_t), \tag{1}$$

where $FE : I\!R^n \longrightarrow I\!R^m$ and $f_t(i)$ is the i-th feature value referred to time instant t, $i = 1 \ldots m$.

As an extraction function, FE, shrinks the amount of data, retaining the essential information of sensor data. For that reason,

$$FE^{-1}(f_t) \supset o_t,$$

which means that different observation vectors could result on the same feature. When this convergence on the same feature occurs, it is important to identify if the observations were acquired in the same place, or in places where it is not important to distinguish the features.

Edges and Hough-Transform

As described in [13], the image dependencies due to the lighting geometry and illuminance, mainly in outdoor environments, require a colour image normalization procedure. This drawback points towards the edges extraction. Referring to the bibliography, a couple of authors present different approaches using edges extraction, for environment representation and robot navigation [6], [7], [8], [9] and [5].

To extract edges from an image, it is necessary to apply a specific filter (Sobel, Prewitt, Roberts, Gaussian, or other). However, in outdoor environments, where the scenario is unstructured, the edges may present noisy information. According to this fact, it is necessary to remove or, at least, reduce the superfluous data. As proposed by [10] and [12], the straight lines are important geometric information from the images, mainly the vertical ones. A straight line is defined by (d, θ),

$$x \cos\theta + y \sin\theta = d,$$

where (x, y) are the coordinates of an image pixel. Consequently, a powerful technique is used: the Hough Transform (HT) to the edges, [11]. The result is an histogram of straight lines for different directions, as shown in Figure 1 for real acquired outdoor data, where the brightness corresponds to the amount of pixels that belong to a specific line. To select only the vertical edges, the directions chosen are around 0 and 180 degrees, as exemplified in right-bottom image of Figure 1. The k_{edges} straight lines with more pixels (high level on the histogram) are selected and considered as the edges' features extracted from the image, i.e.,

$$f_t = FE(o_t) = \{(r_1, \theta_1), \ldots, (r_{k_{edges}}, \theta_{k_{edges}})\}. \tag{2}$$

Part (b) of Figure 1 shows the result.

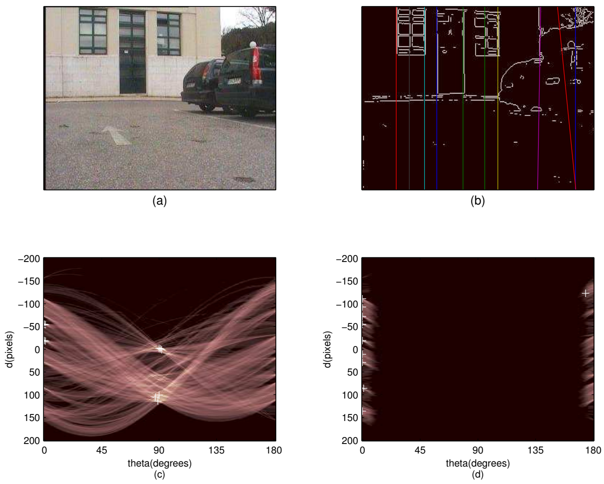

Figure 1: Example of vertical edges detection. (a) - original image; (c) - the Hough Transform of edges; (d) - the same only around vertical directions; (b) - the features extracted: vertical edges

Histogram parameterizations

Even with the light and geometric dependencies, the colour information is still an important source of information. Applying a normalization procedure, as suggested by [13], or simply, using the HSV colormap in spite of RGB, color histograms are important features. However, histograms provide large amount of information that could be parameterized. We tested the parameterization of Hue and Saturation histograms using polynomial and Gaussian functions. A parameterization of each histogram using a polynomial function of order n requires $n+1$ parameters $(a_0, a_1, \ldots, a_n)$, while by $N(\mu, \sigma)$ requires $2n$ parameters. The parameterization error is evaluated using the square error of the original histograms and the parameterization. Testing in a large amount of images acquired in different places of an outdoor environment, the Gaussian parameterization errors are significantly lower when the number of parameters are larger then 4, as shown in Table 1.

The features extracted are the parameterizations, or more precisely, the means and variances. Equivalently,

$$\begin{aligned} f_t &= FE(o_t) \\ &= \{(\mu_1, \sigma_1)_H, \ldots, (\mu_{k_{histograms}}, \sigma_{k_{histograms}})_H, \\ &\quad (\mu_1, \sigma_1)_S, \ldots, (\mu_{k_{histograms}}, \sigma_{k_{histograms}})_S\}. \end{aligned} \tag{3}$$

where H and S correspond to the Hue and Saturation respectively.

Histogram 2D and image segmentation

♯ of parameters	Gaussians		Polynomials	
	H color	S color	H color	S color
2	3214	1087	3247	1889
4	2010	830	3092	1505
6	1937	534	2720	1112
8	714	478	2406	867

Table 1: Comparison of the parameterization error using Gaussian and polynomial functions

Using histograms it is possible to identify regions on the image with similar colors performing the bi-directional histogram along Hue-Saturation colors. Based on the histogram, the k_{hist2D} most significant colors $(Hue, Saturation)$ are selected. For each significant color we defined a region on the image, explained as follows. The smallest boundary-box that fits all the pixels with the same color, defines a region. The features extracted from each boundary-box are the width, the height, the amount of pixels and the variance.

$$\begin{aligned} f_t &= FE(o_t) = \{(box_1), \ldots, (box_{k_{hist2D}})\} \\ &= \{(width, height, pixels, variance)_1, \ldots \\ &\quad (width, height, pixels, variance)_{k_{hist2D}}\} \end{aligned} \tag{4}$$

The position of the boundary-box on the image is not recorded, since it is much dependent on the point of view, [16].

PCA and ICA

A common approach used by the authors of [14] and [17] to extract the essential information from images is based on the Principal Component Analysis (PCA). A similar technique, where the components are orthogonal, is the Independent Component Analysis (ICA), cited in [15]. Both techniques consist in extracting a base, $B = \{B_1, B_2, \ldots, B_{k_{comp}}\}$ from a training set of images. The projection of the training set into each base B (for PCA or ICA) provides different energy distribution. The PCA results condenses the energy into the first components (usually the 2 first retain more than 90%).

Given the size of images the author of [15] proposes to optimize the implementation dividing the images into sub-images. This approach is also interesting, since the original images present common areas (for instance: the ground, the sky), as illustrated in Figure 2 acquired inside the campus of *Instituto Superior Técnico.*

Therefore, the basis depends on the number of components and on the number of sub-images. This relation is not linear, as exemplified by the results shown in Table 2. This table describes an example of reconstructed images using PCA (left-side) and ICA (right-side), with 5,10,15,20,25 components. The columns correspond to the sub-divisions of the images (1-no division, 4,16-divides the image into 4 and 16 sub-images respectively, as illustrated in Figure 4). The error is an average the for all pixels (each pixel changes between 0 and 255). When the number of components increases, the error decreases. For instance, the 1st column of PCA (or ICA), the reconstruction error is zero when the number of components is larger than 12, since the size of the training set is 12 images. However, the reconstruction using 5 or 10 components and images divided into 4 sub-images provides an error larger than when using images divided into 16 or not divided. This is inverted for more components. It

Figure 2: A training set of images.

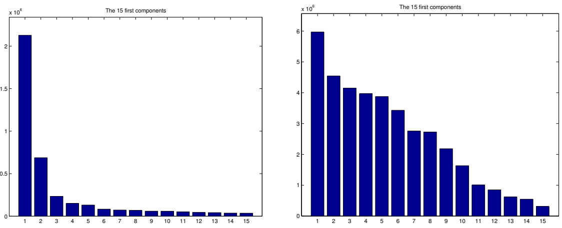

Figure 3: L_2 norms of the basis functions using PCA (Left) and ICA (Right)

is related to the images when divided into 4 parts, the sub-images coincides with the ground or the sky or the buildings.

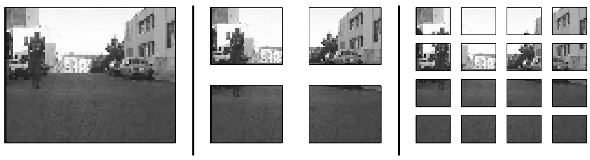

Figure 4: Left to right: The entire image; Image divided into 4 sub-images; Divided into 16

The features are the projection of the observed images, o_t, on the base, B, or equivalently:

$$f_t = FE(o_t) = \{< o_t, B_1 >, \ldots, < o_t, B_{k_{comp}} >\}. \tag{5}$$

Both techniques can be applied to the image to each color RGB or HSV, where only HS is more robust, as explained before.

2.2.2 Feature Selection

As soon as features are extracted it is necessary to define a selection criteria. The quality of a feature has to be analyzed along two criteria: time/space and correlation with the other features. The selection along these lines can be illustrated by an example. Let us assume that there are two type of features: "colors" and "geometric forms", and that the robot is navigating along three distinct places. If all the places are identified by the same color, this renders the feature "color" useless, independent of the "geometric" information. If two places are "red" (the same value for the feature "color") and the third place is "blue", the

♯ of com-	PCA			ICA		
ponents	1	4	16	1	4	16
5	6.6	7.3	6.1	11.5	12.2	8.9
10	1.7	5.7	5.1	2.3	11.0	8.7
15	0	4.4	4.5	0	10.4	8.5
20	0	3.2	4.1	0	8.9	8.3
25	0	2.0	3.8	0	8.0	8.1

Table 2: Comparison of image reconstruction using PCA and ICA

feature "color" can identify some places, but the ambiguity is still present. In this case, if the geometric form is the same for the "red" places, but different for the "blue" place, it means that the two features are redundant (the correlations between features is too high).

The main goal of feature selection is the following: given a training set of observations, find the best features that minimizes the ambiguity and remove the redundant ones. This is the current topic of work within topological navigation at RESCUE project. The future research also includes a feature extraction procedure based on textures extraction using Gabor Filters and/or Nonlinear Operator [18], [3], [14].

3 Distributed Planning for a Multi-Robot Rescue Team

Typically, a rescue operation within a situation of catastrophe involves several and different rescue elements (individuals and/or teams), none of which can effectively handling the rescue situation by itself. Only the cooperative work among all those rescue elements may solve it. Considering that most of the rescue operations involve a certain level of risk for humans, depending on the type of catastrophe and its extension, it is understandable why robotics can play a major role in Search and Rescue situations (SaR), especially teams of multiple heterogeneous robots.

The overall goal of the RESCUE project is to develop a robotic team, constituted by more than one robot, capable of autonomously handle a rescue operation. This project can be seen at different levels of abstraction, such as a technological level (e.g., hardware development), a control level (e.g., motor control), a robot navigation level, and a task planning level, if an individual robot is considered. If we are to consider a team of robots, new levels must be added, for instance a level of robot cooperation and a level of mission management. At these levels, the objectives are making robots cooperate to fulfil their common goals, both through cooperative planning and cooperative execution. The RESCUE project aims at the development of an integrated approach to most of referred levels of abstraction, initially for a simplified rescue scenario and a team of two robots (an aerial one and a terrestrial one).

The work developed on multi-robot planning is mainly focused on the problem of distributed task planning for a team of heterogeneous robots. However, all considerations related with technology and utilization of real robots was not an issue in this work. So our rescue team is composed of agents, virtual entities interacting within a simulated environment and capable of some intelligent actions, both individual and cooperative.

The problems of task planning, task allocation and cooperative execution is dealt with mainly in the areas of Distributed Artificial Intelligence or Multi-Agent Systems. The main questions to be answered when solving these problems are:

Selection of goals and their allocation among agents Given a non-empty set of (rescue) objectives, agents must be capable of selecting the right sequence of goals to be fulfilled and distributing these goals among them.

Task planning restricted by the agents actions Given a particular goal, an agent must design a plan of actions that enable it to achieve the goal. Planning in this context means finding a sequence of actions that takes the agent from an initial state of beliefs to another state where a certain set of beliefs is included. This plan can include not only the actions the agent has but also all actions it knows other agents have. Therefore, plans tend to become non linear, i.e., there are actions to be performed in parallel by different agents.

Plan execution Besides ensuring that all pre-conditions for the whole plan and each one of its actions are met, some actions must be synchronized among agents.

Resources management One of the main resources in rescue operations is time, in the sense that timing is usually vital for the rescue success, not only the plan execution time but also the planning time. So a tradeoff is needed between the quality of plans and time to determine them.

Failures recovery The problem here is to decide what to do when premises for the plan being executed change. Agents must react promptly to changing conditions not only by deciding what to do next, either adapt the current plan or re-plan, but also in order to bring the team of robots, if that is the case, to a common and consistent state of beliefs.

Distributed planning One of the advantages of having several robots is also the possibility of dividing the computational needs among them. For instance, instead of performing task planning in only one robot or computer, one might divide it between two, three or even more robots. The problem of course is to decide who and what each one will plan.

Coherence and cooperation A known problem in multi-robot/agent systems is the possibility that one agents actions could invalidate other agents actions, due to, e.g., non-shared resources. So it is necessary to ensure at execution time the coherence of plan being executed.

Communication Obviously, in a multiple robot scenario, communication is always a relevant issue, both because it is limited and the agent must decide what to communicate.

Given all the problems described above, the project work has focused mainly on the problems of task planning and task allocation in a multi-robot rescue system, assuming that teamwork (i.e., cooperative tasks) plays an important role on the overall planning system.

An agent architecture has been developed, inspired on a Belief-Desire-Intention (BDI) architecture [19], considering that each agent interacts with others in the same rescue scenario, with the same interface and ontology. Moreover, the proposed architecture takes into account issues as agent heterogeneity, failures recover, cooperation, to name but a few. Besides that, agents equipped with this architecture are prepared to act in a non deterministic environment (where its state could change without any agent action), incomplete (meaning that only information agents have is acquired by their sensors which provided only incomplete data about the environment state), dynamic (meaning that planning decisions made for a certain environment state could be invalid when they are executed, claiming for some re-planning).

Since teamwork is a key aspect of this work, agents need to negotiate the execution of certain actions, either because an agent does not have the right skills to do it, or it evaluates that another agent could do it better (with a lower cost). To implement this a Contract-Net system [20] was developed and integrate in the agent architecture. This system allows agents to propose and negotiate contracts with other agents, and gives the necessary guarantees for maintaining signed contracts consistency (i.e., if an agent cannot fulfil a contract it must inform others involved in that contract).

The main decision process, the planner, was implemented based on a Hierarchical Decomposition Partial Order Planner (HDPOP) approach, with an important extension, the possibility to handle (plan) the resources needed for each of the tasks [21, 22]. The planner was developed using the STRIPS language and is supported on a variation of the well-known A^* search algorithm, the Iterative Deepening A^* (IDA^*).

To experiment and evaluate the proposed planning system, a simplified version of a rescue simulator was also developed. This simulator allows to create virtual rescue scenarios where rescue teams should face building and forest fires, civilians trapped in collapsed buildings, and roads blocked. The rescue teams are composed of aerial and land robots, with different skills. The former could perform a survey of the affected region (for instance, by defining a topological map and send it to the other robots, namely the land ones). They are also capable of transporting victims to rescue spots. On the other hand, the latter are endowed with first aid resources and have the autonomy to decide if the victim might be transport by air (in which case it contracts an aerial robot to take care of that transport).

Although this work did not cover all the problems mentioned earlier, the results obtained show that a distributed approach to a rescue problem is clearly an interesting solution when compared with a centralized one. One might lose some quality of the planning solutions, but gains more flexibility, redundancy and the possibility of parallelizing the planning process. One key word emerging from this work and its results was *delegation*, meaning that agents should delegate as much as possible given other agents skills, particularly whenever planning is concerned [23].

4 Software and Functional Architectures

The software architecture developed for the RESCUE project is supported on agent-oriented programming concepts that provide a systematic method for task design, task planning, task execution, task coordination and task analysis for a multi- robot system. An application program interface (API) was implemented and is described in Appendix B, where the work done is fully described. This reference guide is targeted for researchers and students working on the RESCUE project, as well as to future users of the architecture, extendable to other projects.

The conceptual model of the agent-based software architecture includes different types of agents that can be combined both hierarchically and in a distributed manner. The architecture supports information fusion between several sensors and the sharing of information between the agents by a Blackboard (a distributed structure that gives support to the data exchange between the Agents), and is geared towards the cooperation between robots. Agents are generically organized hierarchically. At the top of the hierarchy, the algorithms associated with the agents are likely to be planners, whilst at the bottom they are interfaces to control and sensing hardware. The planner agents are able to control the execution of the lower level agents to service high-level goals. The latter can be distributed across several processors and/or robots. To offer platform independence, only the lowest level agents are specific to the hardware, and these have a consistent interface for communication with the planning agents that control their execution. The elements of the architecture are the **Agents**, the **Blackboard**, and the **Control/Communication Ports**. Agents communicate with each other through *control ports* and with the blackboard through *data ports*. The latter is effectively another means of sharing information among the agents.

In Robotics research and development, much time and resources are consumed in system design, system calibration and system analysis. A well-designed architecture targets the support and speed-up of these development phases. Usually, properties such as system distribution and concurrency are relevant during the mission execution, since they provide better resource allocation and robustness. Under this architecture, a different execution mode exists for each development phase of a multi-robot system. Five execution modes are available for each of the elements described in the previous section:

Control Mode that refers mostly to the run-time interactions between the elements.

Design Mode

Calibration Mode

Supervisory Control Mode

Logging and Data Mode .

References

[1] P. Lima, M. Isabel Ribeiro, Luis Custodio, Jose Santos-Victor. "The RESCUE Project - Cooperative Navigation for Rescue Robots", Proc. of ASER'03 - 1st International Workshop on Advances in Service Robotics - Bardolino, Italy, March 13-15, 2003.

[2] A. Vale, M. I. Ribeiro. "Environment Mapping as a Topological Representation", Proc. of the 11th International Conference on Advanced Robotics, Coimbra, Portugal, vol. 1, pp. 29-34, 2003.

[3] S. Grigorescu, N. Petkov, P. Kruizinga. "Comparison of Texture Features Based on Gabor Filters", IEEE Transactions on Image Processing, **11(10)**, 2002.

[4] A. Vale, M. I. Ribeiro. "A Probabilistic Approach for the Localization of Mobile Robots in Topological Maps", Proc. of the 10th IEEE Mediterranean Conf. on Control and Automation, Lisboa, Portugal, 2002.

[5] Nikos Vlassis, Yoichi Motomura, Isao Hara, Hideki Asoh, Toshihiro Matsui "Edge-based Features from Omnidirectional Images for Robot Localization" *Proceedings of the 2001 IEEE Conference on Robotics and Automation, Seoul, Korea, May 2001*

[6] M. Mata, J. M. Armingol, A. de la Escalera and M. A. Salichs "Using Learned Visual Landmarks for Intelligent Topological Navigation of Mobile Robots" *Proceedings of the IEEE International Conference on Robotics and Automation, September 2003*

[7] Pierre Lamon, Illah Nourbakhsh, Björn Jensen, Roland Siegwart "Deriving and matching image fingerprint sequences for mobile robot localization" *In Proceedings of the IEEE International Conference on Robotics and Automation, Seoul, Korea, May 2001*

[8] Mark Fiala, Anup Basu "Robot Navigation Using Panoramic Landmark Tracking" *Proceedings of Vision Interface pp. 117-124, Calgary, Canada, May 2002*

[9] Xiang Sean Zhou, Thomas S. Huang "Edge-Based Structural Features for Content-Based Image Retrieval" *Pattern Recognition Letters, April 2001*

[10] Hyun-Deok Kang, Kang-Hyun Jo "Self-Localization of Mobile Robot Using Omnidirectional Vision" *Proceedings of the 7th Korea-Russia International Symposium, KORUS 2003*

[11] Klaus Hansen, Jens Damgaard Andersen "Understanding the Hough transform: Hough cell support and its utilization" *Journal IVC, vol. 15, pp. 205-218, 1997*

[12] José Santos-Victor, Alexandre Bernardino "Vision-based Navigation, Environmental Representations and Imaging Geometries" *Robótica, n. 47, Second Trimester, Portugal, 2002*

[13] Graham D. Finlayson, Bernt Schiele and James L. Crowley "Comprehensive Colour Image Normalization" *Lecture Notes in Computer Science Journal, 1998*

[14] I. T. Jolliffe "Principal Component Analysis" *Springer-Verlag, New York, 1986*

[15] Thomas Wachtler, Te-Won Lee, Terrence J. Sejnowski "The Chromatic Structure of Natural Scenes" *Journal of the Optical Society of America, A , Vol 18 (1), 65-77, January 2001*

[16] W. Kasprzak, W. Szynkiewiez "Using Color Image Features in Discrete Self-Localization of a Mobile Robot" *International IEEE Conference on Methods and Models in Automation and Robotics, vol.?? no.2, pp.1101-1106 Szczecin-Miedzyzdroje, Poland, August 2003*

[17] Frank Wallner, Bernt Schiele, James L. Crowley "Position Estimation for a Mobile Robot From Principal Components of Laser Range Data" *Proceedings of IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998*

[18] Simona E. Grigorescu, Nicolai Petkov and Peter Kruizinga "Comparison of Texture Features Based on Gabor Filters" *IEEE Transactions on Image Processing, vol.11, no. 10, October 2002*

[19] M.E. Bratman "Intentions, Plans and Pratical Reason" (2nd edition) Addison-Wesley, 1987.

[20] M. D'Inverno and M. Luck "Normalizing the Contract-Net as a Goal Directed System" *in MAAMAW'96, Springer-Verlag, 1996*

[21] K. Decker and V. Lesser "Designing a Family of Coordination Algorithms" U. Massachussets Computer Science Technical Report 94-14, Aug 1995

[22] E. Durfee and V. Lesser "Generalizing the Partial Global Planning Algorithm" U. Massachusetts, June 1993.

[23] Dinis Pereira, Rodrigo Serafim, "Planeamento Distribuido para um Sistema Multi-Robot", Final Project Report, LEEC/IST, 2003