# Multi-agent system for fast deployment of a guide robot in unknown environments

A. Canedo-Rodriguez, V. Alvarez-Santos, C.V. Regueiro, X. M. Pardo and R. Iglesias

***Abstract*—Nowadays, deploying service robots and adapting their services to a new environment is a task which might require several days. This is an important problem of robotics in general, but specially when the goal is to bring robots to our everyday life. In this paper we present a multi-agent intelligent space, which consists on intelligent cameras and autonomous guide robots. The deployment of the system does not require expertise and can be done in a short period of time. The cameras detect situations requiring the robots' guiding services, inform the robots accordingly, and support the robots navigation towards the goal areas, without the need of a map of the environment. An example of these situations requiring the robot guide service could be a group of persons entering a museum. In this sense, we also present an adaptive person follower behaviour intended to be the basis of a route learning process, necessary to offer the guide service.**

***Index Terms*—Guide robot, multi-camera networks, intelligent space, person following, feature weighting.**

## I. Introduction

ROBOTS are expected to become part of our everyday life, either as assistants, house appliances, collaborating in the care of elderly people, etc. These service robots need to be able, on the one hand, to work on complex and dynamic environments and, on the other hand, to offer advanced capabilities useful to people. Despite the increasing demand for personal robots able to educate, assist, or entertain at home, or for professional service robots able to sort out tasks that are dangerous, dull, dirty, or dumb, there is still an important barrier between the latest developments on research robots and the commercial robotic applications available. The deployment of robots and the adaptation of their services to different environments usually require the tuning of their software and hardware to the particular conditions of each environment. This task is not trivial and might require several days in most cases, which makes the process of bringing the robots to everyday life extremely inefficient, specifically when the robots are targeted for short term operation, as is typical in sporadic events.

Research groups working on robotics receive numerous requests to take the robots to social events (museums, forums, etc.). However, conducting these demonstrations in different places takes a big effort, unless the robot runs the same programs under boundary conditions known a priori. A fast and easy deployment of robots in new areas is necessary to get robots operating outside research centres and beyond the continuous supervision of roboticists. In this sense, we are currently involved in the development of general purpose guide robots as part of the research project "Intelligent and distributed control scenario for the fast and easy deployment of robots in diverse environments". Through this project we try to develop robots which are able to participate in different social events (e.g. schools, museums, forums, conferences, etc.), providing useful information to visitors. Two of our goals within this project are:

- *Development of deployment strategies which reduce the amount of time required for putting our robots in operation in different environments and conditions, and also the expertise required to carry out this deployment.* Robots must be capable of being installed and put into operation within very short periods of time. Robots should be able to work in these environments without prior knowledge about them (e.g. metric maps), or without requiring significant changes on them. Finally, the deployment of the robots should not require expert knowledge (i.e. it must be as automatic as possible), prioritizing online adaptation and learning over hard-coded and pre-tuned robot behaviours.
- *Improve the quality of the guide robot and the people's opinion on them, providing them with sophisticated capabilities.* Our guide robots should be able to offer the possibility of showing different routes to those people attending an event, or to help them to reach a specific location in the environment. Nevertheless, this requires certain knowledge of these routes or places of general interest. Therefore, our robots should be able to learn routes and points of interest along these routes, by simply following a staff member working in the same building where the robot is. On the other hand, our robots should not only provide these services on people's request, but they also should be able to identify situations in which their services would be useful. Finally, our system must be reliable, ensuring the continuity of correct service as much as possible (i.e. our robots should be able to recover and learn from failures when they occur).

Most of the research done so far has focused on self-contained, stand alone robots that act autonomously, doing all the sensing, deliberation, and action selection on board. Nevertheless, in these systems, the robot sensing is restricted to the capabilities of its on-board sensors, which limits the

A. Canedo-Rodríguez, V. Álvarez-Santos, X. M. Pardo and R. Iglesias are with the Centro de Investigación en Tecnología de la Información de la Universidade de Santiago de Compostela (CITIUS). Campus Vida. Universidade de Santiago de Compostela.
E-mail: adrian.canedo@usc.es
C. V. Regueiro is with the Department of Electronics and Systems, of the Universidade de A Coruña.

ability of the robots to respond to events and to learn from its perceptions and actions. In this sense, technologies from the fields of ubiquitous and pervasive computing, sensor networks, and ambient intelligence could be integrated with robotics, providing a new way to build intelligent service robots that opposes the idea of stand-alone robotic platforms, namely ubiquitous robotics [1]. Intelligent sensor networks, which can be spread out over wider areas, will provide an invaluable source of information from beyond robots' immediate surroundings, allowing the robot to respond to a wide range of events, and making it look like it has initiative. Moreover, intelligent sensors could also enhance robots' local perceptions, supporting the movement of the robot in the environment. For example, a camera might detect a group of people entering a museum, and alert the robot about its presence, so that the robot might approach this group of people, and offer them its services. Finally, these intelligent sensor networks might be used to build high level symbolic representations of their location on the environment, removing or relaxing the need of detailed maps.

In this paper, we present the first stage of the ongoing research aimed at building an intelligent deployment strategy. The deployment of robots consists on creating a multi-agent distributed network of intelligent cameras and autonomous robots. The cameras are spread out on the environment, so that they can detect events (such as groups of people) which might require robots' services, they can inform the robots about these events, and they can also support the movement of the robots in the environment. The robots, on the other hand, navigate safely towards those areas of interest detected from the cameras, to offer the guidance along different routes. To this extent, in this article we also present an adaptive person following behaviour, able to work on crowded environments and/or under challenging illumination conditions

Section II provides an overview of the previous works concerning service robots, focused on their capability to be deployed fast and easily. Section III provides a general overview of our system. Section IV describes the tasks performed by the camera network. Section V describes the basis of the route learning process. Finally, experimental results and conclusions will follow.

## II. Previous work

Over the last two decades, there have been remarkable examples of social robots, mainly of informational and guiding type, able to work on social events, like Rhino (1995) [2], Minerva (1999) [3], Tourbot and Webfair (2005) [4], and Urbano (2008) [5]. Both their quality and their deployment times have improved importantly: from 180 days of installation required by Rhino [4], to less than one hour required by Urbano for a basic installation [5]. On the contrary, major ubiquitous robotics projects have improved the quality and applicability of their systems, but the reduction of their deployment time has not received much concern.

Lee et al. (Hashimoto Labs.) [6], proposed the concept of Intelligent Spaces, as rooms which perceive and understand what is happening in them, and which perform tasks for humans. They proposed to distribute sensors (typically cameras) with processing capabilities and used them for supporting robots' navigation [7], [8]. Similarly, Intelligent Spaces have been applied successfully on the tasks of robot localization and tracking from single [9] and multiple cameras [10], and also on the task of robot navigation [11], [12], [13]. Regretfully, these proposals do not deal with the problem of easy and fast deployment in social environments, and they are not likely to be appropriate for it: all of them have been designed and validated to work in small places (less than 50 square meters), typically requiring a great number of highly coupled cameras, relying on a centralized processing and coordination, and wired communications.

The Japan NRS project, and the URUS project are a step closer to our philosophy. The NRS project focuses on user-friendly interaction between humans and networked devices (informative robots, distributed sensors, etc.), and they demonstrated the use of their systems in large real field settings, such as science museums [14], and shopping malls [15], during long term exhibitions. In the same line, the URUS project [16] proposes to deploy a network of robots, sensors and other networked devices in wide urban areas (experimental setup of 10000 $m^2$), to interact with the people and to perform different tasks (information tasks, transportation, surveillance, etc.). Regretfully, none of these projects consider the efficiency of the "deployment" phase, probably because their systems are intended for long term operation in the same places, where the costs of the deployment are not critical.

To the best of our knowledge, the projects above are the most representative in the context of our work, and none of them target the problem of the efficient robot deployment in unknown environments. Thus, a solution to get robots out of the laboratories to work in different environments within reasonable time and cost is still to be proposed.

## III. System description

Our goal is to design and develop guide robots able to offer information and routes to the visitors of different events, in which they will work. We want these robots to be flexible to work in different environments, requiring the minimum adaptation time, effort and expertise in order to get them working, avoiding any software or hardware tuning between environments. Our system would detect groups of visitors, and the robot would navigate safely within the environment towards them, to offer them guiding services along pre-learnt routes. Also, these guide robots should not require prior knowledge of the environment (e.g. metric maps), since they should adapt as fast as possible to it. In the absence of this knowledge, the users of our system will not use a map to teach the robots the routes of interest, but they will make the robots to follow them along the real routes. Therefore, our robots will also need to be able to follow a human teacher through the environment. Usually this environment is also crowded with other people, but this should not confuse or disturb the robot.

With this system in mind, we have identified two different and important phases: deployment phase and use phase. The first one is the configuration phase, consisting on deploying

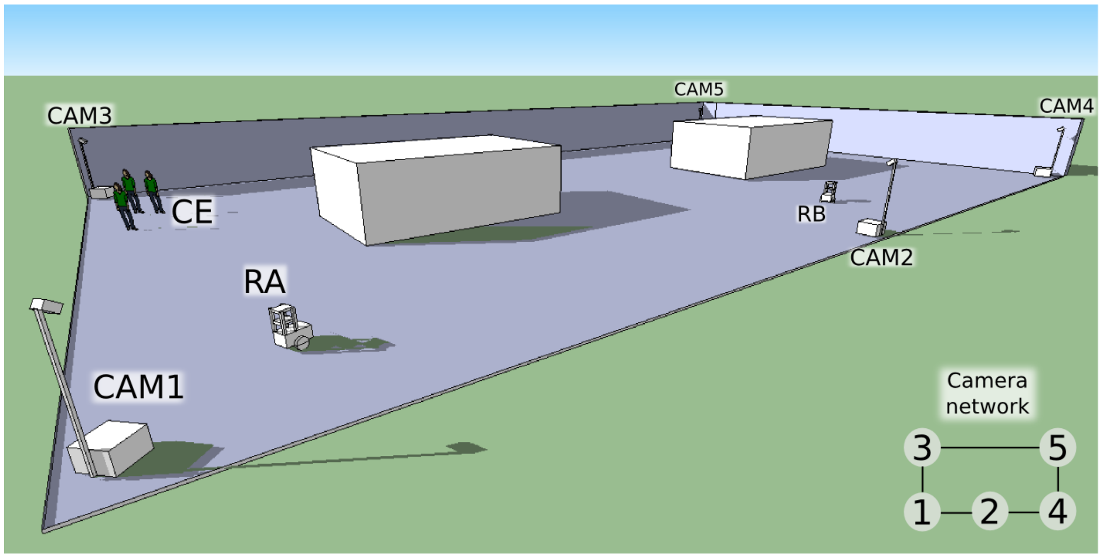

Fig. 1. Example of the deployment of the multi-agent system: camera-agent 3 (CAM3) is detecting a Call Event, while robot-agent A (RA) is being sighted by camera-agent 1 (CAM1) and robot-agent B (RB) by camera 2 (CAM2).

all that it is necessary for the system to work, and on teaching the routes to the robot. We require this configuration phase to be as short and fast as possible, and it also must be easy to accomplish it for anybody (even non expert users). In the second phase, the system should detect groups of visitors that might require the robots' guiding services (Figure 1). Robots should navigate towards this groups of people to offer them the possibility of following routes that have been previously learnt. In short, in the first phase we would deploy and configure the system, while in the second phase the system would actually give the services for which we designed it.

To accomplish these tasks, we propose a multi-agent system such as the one illustrated in Figure 1, which consists on two main elements: a) an intelligent control system formed by camera-agents spread out on the environment (CAM1 to CAM5 in the Figure), and b) autonomous guide robots navigating on it (RA and RB in the Figure). Basically, the cameras detect Call Events (CE in the Figure) within their Fields of View (FOVs from now on): typically, groups of visitors. Then, the cameras support the robots to navigate towards the Call Events, so the need of a map is avoided. Once the robots arrive at the destination, they offer information or guiding along routes of interest to these people. The agents coordinate in a fully distributed fashion, based on local interactions. This allows us to introduce more agents in the system without major changes, favouring scalability and robustness. In Section III-A and III-B we describe both agents in more detail.

### A. *Camera-agents*

Each camera-agent consists on an aluminium structure like the one represented in Figure 2, which is easy to transport, to deploy and to pick up. As we can see, the aluminium structure is made up of a box, which contains a processing unit, a WIFI Access Point and power supplies, and a mast which holds one or more video cameras (one for each camera-agent).

On the other hand, the software consists on four concurrent modules, responsible for carrying out the tasks corresponding to these agents (Section IV):

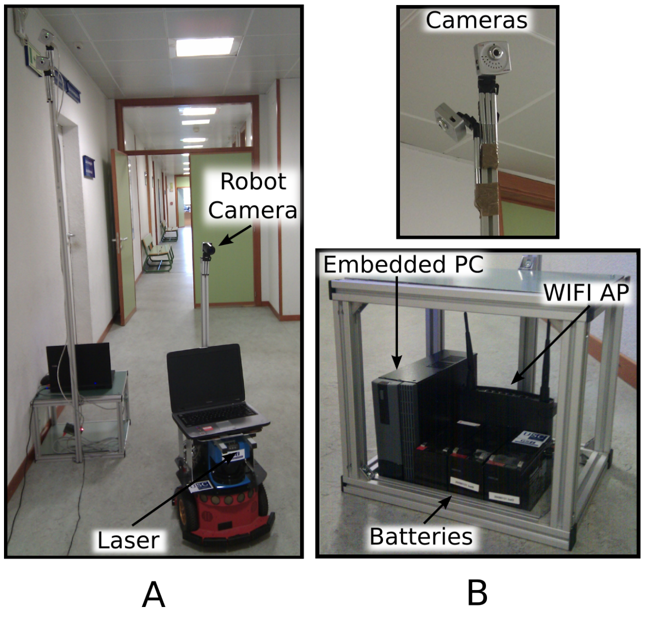

Fig. 2. A) Camera-agent (left) and robot-agent (right). B) Hardware elements that make up each camera-agent.

- *Vision Module*: responsible for the detection and tracking of robots and call events.
- *Network Module*: responsible for wireless communications.
- *Call Module*: responsible of notifying robots about call events present in its FOV.
- *Neighbourhood Module*: responsible of detecting neighbourhood relationships with other cameras by detecting common parts among their FOVs. As we will describe later, by establishing local neighbourhood relationships and by interacting locally with their neighbours, the

cameras will be able to form routes among them, and to support the navigation of the robots.

### B. Robot-agents

We work with wheeled robots like the Pioneer 3DX, equipped with a laser scanner and a video camera (Figure 2). The software of the robot is divided in two concurrent modules: a *Network Module* for communications and a *Control Module* for navigation. The Control Module is based on the classic but robust Potential Fields Method: the robot moves towards a goal position which exerts an attractive force on it, while the obstacles detected by the laser scanner exert repulsive forces, so that the robot avoids colliding with them. As we will see later, this goal and therefore the attractive force can be either provided by cameras when supporting robots' navigation, or by the users of the system when the robot is learning a tour route by following any of them.

## IV. Deployment of an intelligent network space

As we have stated before, camera-agents are responsible for detecting events requiring robots' presence, and for supporting robots' through the commitment of their duties. In order to fulfil these objectives, the cameras perform the following low-level tasks: dynamic neighbourhood detection, distributed route planning, and support to robot navigation. First, upon distribution in the environment, each camera discovers who are its camera neighbours (*dynamic neighbourhood detection*). When a robot is required to go from one camera to another, the cameras execute a distributed process based only on local interactions with their neighbours, to discover all the possible inter-connection routes (*distributed route planning*). Finally, each camera supports the robot's movement towards the next neighbour camera on the route (*support to robot navigation*). We would like to remark that neither a global camera topology representation, nor the calculated routes are stored anywhere: each camera keeps only information about its neighbour cameras. This will be explained in the following sections.

### A. Dynamic Neighbourhood Detection

The correct functioning of the solution we propose requires each camera being aware of which of the other cameras in the intelligent network are its neighbours, and which parts of their FOVs overlap (this overlapping FOVs will be called Neighbourhood Regions from now on). We would like to remark that our system does not require the cameras to know the relative poses (position and orientation) of their neighbour cameras, thus we avoid the introduction of complex matching techniques [17]. Instead, in our system, each camera determines which cameras are their neighbours by detecting simultaneous high-level events, like the detection of a robot from several cameras. Whenever a camera detects a robot, it stores its position and broadcasts it to all the other cameras. Periodically, each camera checks whether the detection of the robot is taking place simultaneously with the detection of the same robot but from any other camera, in which case a neighbourhood relationship will be established.

With this automatic process, the system avoids the need of a metric map of the environment, or even the need of pre-installed knowledge about the distribution of the cameras. Moreover, since the neighbourhood relationships can change dynamically (e.g. if some camera is moved or if it stops working), the cameras are continuously updating this neighbourhood information to adapt to eventual changes, without requiring users to re-configure them.

### B. Distributed Route Planning

A route is an ordered list of cameras through which the robot can navigate. Basically, the robot will go from the FOV of one of the cameras to the FOV of the next camera on the route, without needing metric maps of the environment. These routes will be generated on demand as a result of local interactions among the cameras, without the intervention of any central agent.

We will explain the route planning process through an example, depicted in Figure 3. In this Figure, we represent the system as a graph, with cameras as nodes and their neighbourhood relationships as arcs. However, note that this graph is just for illustration purposes, none of the entities in our system handle this global information. In this example, we assume that robot RA is available (willing to accept any call), while robot RB is not. We also assume that camera 1 is detecting a call event (CE), camera 2 is seeing robot RB, and camera 5 is seeing robot RA.

If a camera, like camera 1 in Figure 3-A, detects a call event requiring the presence of a robot, it broadcasts a call to all the robots. If a robot is willing to attend the call, it broadcasts an acceptance to all the cameras, like robot RA does. Then, those cameras which receive this acceptance and which see this robot will forward the message to its camera neighbours, starting a back propagation process to create a route, as camera 5 does in Figure 3-B: through this process, there will be a message being passed from camera to camera, until it reaches the camera which sees the call event. To do so, each camera which receives this message includes its identity in it, and forwards it to its neighbours (except to those through which the message has already passed). In the Figure, it is clear that camera 5 includes its identity in the message and forwards it to its neighbours, camera 3 and 4, which do the same, and so on. Finally, camera 1 receives all the acceptances, so it knows that RA is willing to accept the call, and that it could follow two possible routes: 5-3-1 and 5-4-2-1. After this, this camera would select the route which involves less cameras (5-3-1), and inform robot RA accordingly. It is clear, after this description, that the route planning does not emerge from a globally coordinated process, but from multiple local interactions among agents just handling local information.

### C. Support to Robot Navigation

When a robot is following a route, each camera on the route helps it to move towards the next Neighbourhood Region, so the robot advances in the route. For doing so, each camera keeps information both about its neighbours and about its Neighbourhood Regions (overlapping FOV areas). Also, when

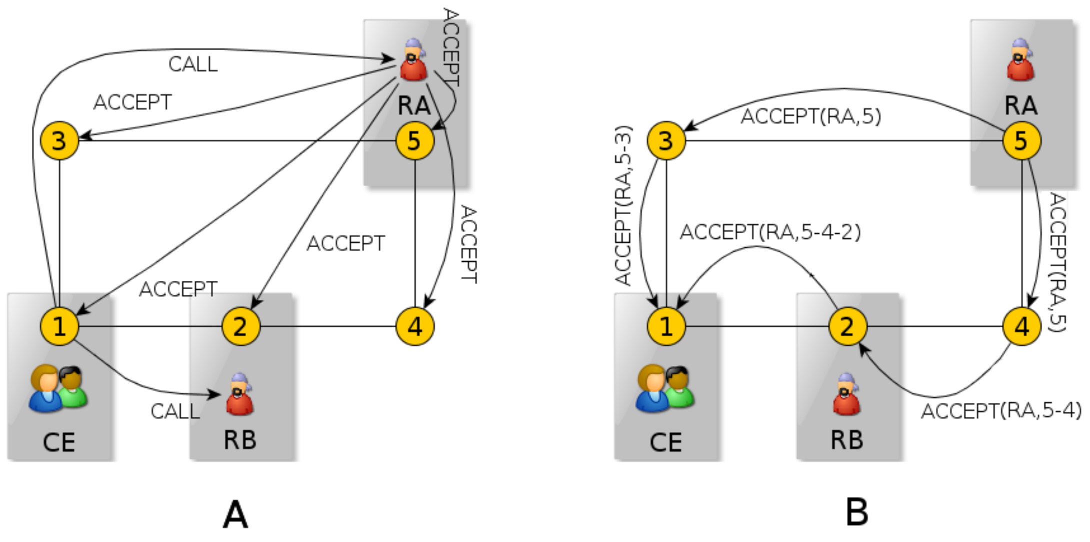

Fig. 3. Distributed route planning procedure. Cameras (1, 2, 3, 4, 5) established their neighbourhood relationships forming a network altogether. A) Call event detection and call for robots. B) Back propagation process for route formation. See the text for a detailed explanation.

a camera detects a robot, it tracks its position and movement direction. This is all the information required by the camera to support robots' navigation.

First of all, if a camera sees a robot, it informs it, after which the robot answers with the route which it is following (in case that it is following any route). Then, the camera informs the robot about the direction that it should follow to get to the Neighbourhood Region shared with the next camera on the route, taking into account the robot's current movement direction. Since the Potential Fields Controller of the robot considers this direction as an attractive force, the robot moves from the FOV of each camera towards the FOV of the next camera on the route, until it reaches the call event.

## V. Route Learning

As we have previously said, one of our goals is to build a guide robot which is able to work in different environments. This robot will offer a route service which can be divided in two parts: route learning, and reproduction of previously learnt routes. So far, our work has concentrated on the first part: route learning.

Teaching a new route to the robot should require neither expertise nor time, since it is a process that must be done every time the robot is brought to a new place. On the other hand, we do not want the users to teach the routes using a map of the environment, since maps are not used by our system. For these two reasons, the robot will learn routes while following a human (the target from now on) who plays the role of the teacher. This process must be safe, avoiding collisions of the robot with the environment, and it must also be robust and adaptive, avoiding mistaking the target for the rest of the people present in the same scene (we call *distractor* to each person moving around the robot and that is not the target). Our goal is to create a robust system for person following and target recognition. Our system must be flexible enough to handle important variations in illumination, scene clutter, multiple moving objects, and other arbitrary changes in the observed scene. On the other hand the person being recognized and followed will not need to wear special clothes or gadgets, thus achieving a more natural human-robot interaction. The presence of distractors moving around the robot might occlude the target, or these distractors might even look similar to the target due to changing light conditions. This is critical, especially when the robot is being taught different routes in crowded environments, where confusing the target might cause the robot to learn wrong routes, and to start over the learning of the route again.

In the next sections we will give a brief description of the person following behaviour that we have designed.

### A. Person Following Overview

An overview of the system we have developed to solve the person following and target recognition problems can be seen in Fig. 4. On one hand, the inputs to the system are the images provided by a camera which is placed on top of the robot, and data from a laser scanner also located on the robot (Fig. 2 A). On the other hand the system provides the position of the target to a robot controller, which will use this information to determine the motor commands that the robot must carry out so that it follows the target and avoids colliding with the environment.

As we can see in Fig. 4 the system we have developed includes two modules which work together to obtain the target's position:

- *The camera module*: The task of this module is to avoid mistaking the target for the distractors. To do this, this module will recognize the target from its torso; in particular it will build, store, and keep updated information about the visual features of the target's torso. An outline of the tasks that this module carries out for every frame acquired from the camera, is: first, the module detects people in the image by using the algorithm developed by Dalal [18].

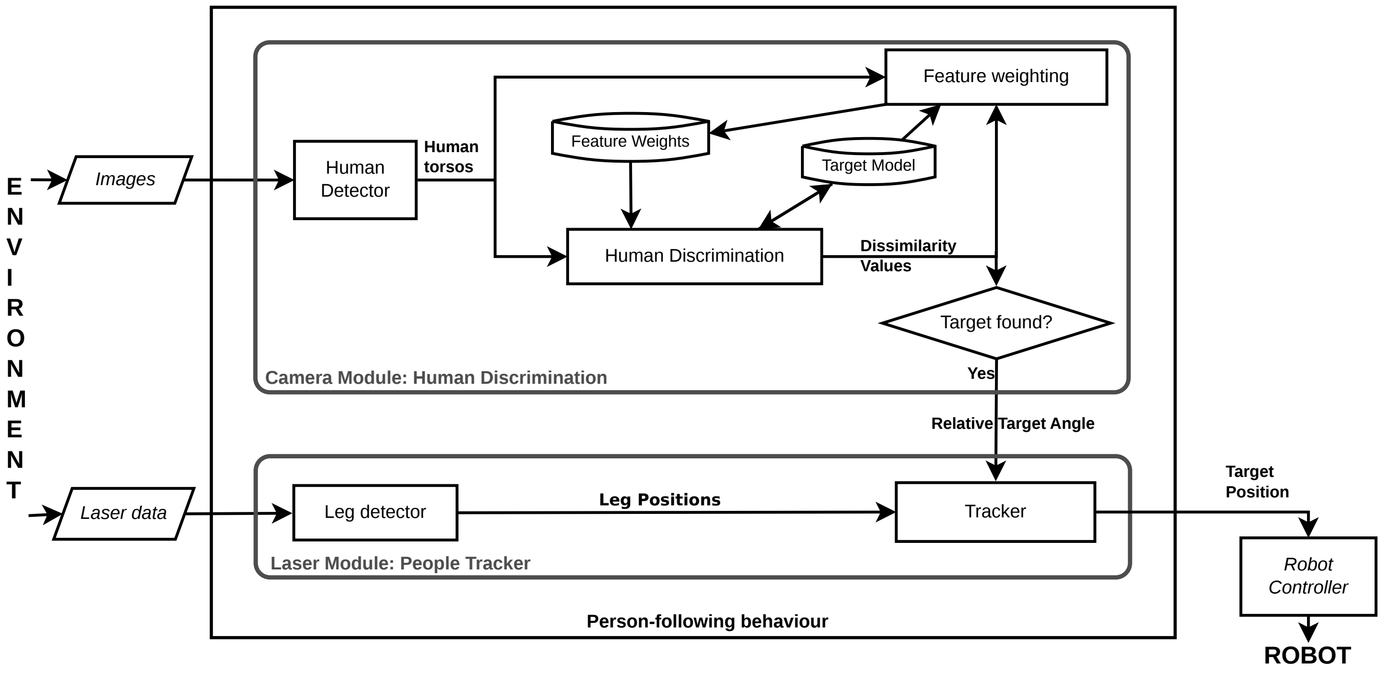

Fig. 4. Schematic representation of the person following behaviour. The camera module identifies the target and sends information about its location to the laser module, which is in charge of tracking its position. In the case that the target is not identified by the camera the laser module can still work.

Using this algorithm it is possible to detect areas in the image in which there seems to be a person. Next, for each one of these areas, this module extracts the visual features of the region that contains the person's torso. These features are then compared with the features of the stored target's torso to determine whether the person in the image is the target or not. Finally, if the person in the image is considered to be the target, the features of the target's torso might be updated using the current detection. On the other hand, this module also determines the angle at which the target is located with respect to the forward direction of the robot, and sends this information to the laser module. When the target is not found no information will be sent to the laser module.

- *The laser module*: The task of this module is to track the robot's target in the course of time. First, this module uses the information provided by a laser scanner located on the robot to carry out a leg detection process. Then, it computes the angles at which the legs are detected with respect to the forward direction of the robot, and sends these relative angles to the person tracker block (tracker in Fig. 4). So far in this description, both sensor modules seem to work separately, nevertheless if the camera provides information about the target, the tracker block will merge it with the information about the leg positions.

According to the previous description, it is straightforward to realize that we use two sensor modalities: a laser scanner, and a camera. The example shown in Fig. 5 summarizes the merging process carried out by the person tracker when it receives information from the camera module. In this figure we can observe three pairs of legs located at the positions $L_1$, $L_2$ and $L_3$. We can assume that these pairs of legs are detected in the laser module, which also computes the angles at which they are detected with respect to the forward direction of the robot: $\theta_{pair1}$, $\theta_{pair2}$ and $\theta_{pair3}$ in Fig. 5. The point $P$ is the last position where the target was detected, and $\theta_{target}$, is the angle at which the camera module found the target, with respect to the forward direction of the robot.

Using this information, the *tracker block* will decide which one of the pairs of legs (pair 1,2 or 3) corresponds to the target. The decision is taken by assigning a probability $p$ to each pair of legs:

$$p = \frac{c(\Delta\theta) + l(x)}{2} \tag{1}$$

where:

1) $c(\Delta\theta)$ is the probability of each pair of legs being the target, according to the camera module:

$$c(\Delta\theta) = \begin{cases} 1, & 0 \leq |\Delta\theta| \leq 4 \\ 1.5 - \frac{1}{8}|\Delta\theta|, & 4 < |\Delta\theta| \leq 12 \\ 0, & otherwise \end{cases} \tag{2}$$

where $|\Delta\theta|$ is the absolute value of the difference between the angle where the camera finds the target ($\theta_{target}$), and the angle where the laser locates the group of legs ($\theta_{pairN}$).

2) and $l(x)$ is the probability of each pair of legs being the target, according to the laser:

$$l(x) = \begin{cases} 1 - \frac{2}{3}x, & 0 \leq x \leq 1.5 \\ 0, & otherwise \end{cases} \tag{3}$$

where $x$ is the distance in meters from the position of each group of legs $L_N$ to the last position where the target has been detected, $P$.

The equation for the $c(\Delta\theta)$ probability was obtained heuristically after checking the behaviour of the robot when it moves

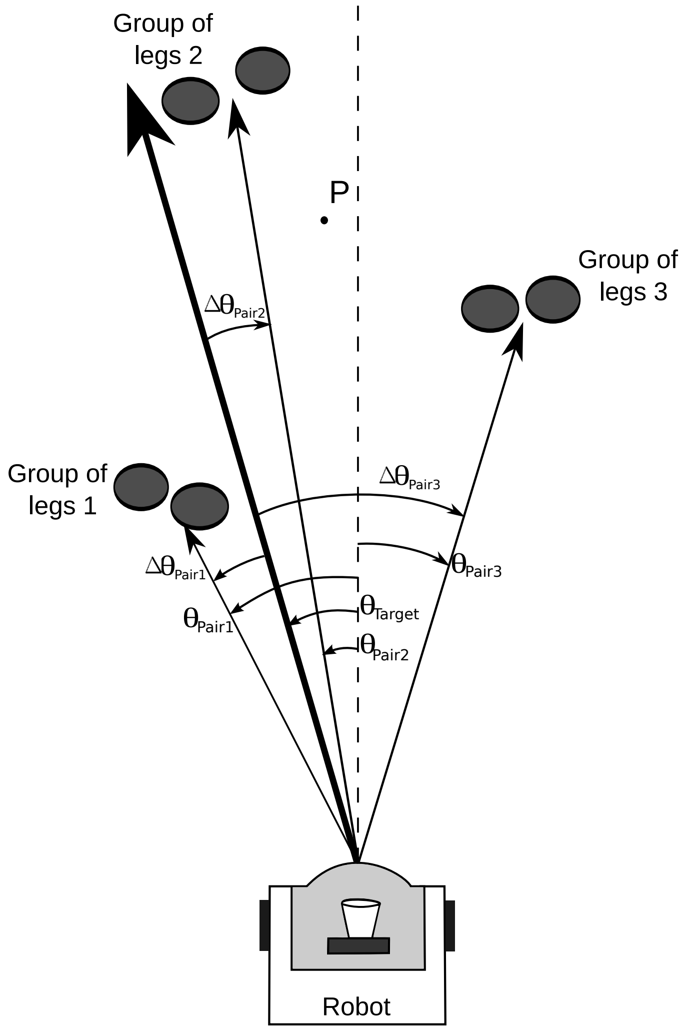

Fig. 5. Example situation where the sensor fusion process takes place.

on an uneven floor. In general we have observed that the vibrations due to this kind of floor might make the camera oscillate in a range of sixteen degrees (eight degrees to each side). This is the reason why $c(\Delta\theta)$ shows a tolerance in that range, i.e., all the pairs of legs which are located within 8 degrees with respect to the direction of the target detected by the camera, should be considered as belonging to the target with a probability higher than 0.5 (Eq. 2).

To build the equation Eq. 3 for $l(x)$ we assume that the walking speed of the target is not higher than 1.7m/s (the average human walking speed is about 1.33m/s). In this case the target should not move more than 0.2m every 100ms (which is the elapsing time between two consecutive laser scans). We also decided to increase the margin of tolerance, due to the high probability of getting noisy measurements; the central position of a pair of legs can be wrong if only one is properly detected, etc. This is the reason why we assume that the position of the target should not change more than 0.75m in two consecutive acquisitions (region around $P$ with a probability of finding the target higher than 0.5, Eq. 3).

We have focused our work on the camera module. In particular we want to find a strategy of combination of visual cues that allow the discrimination of people in different environments, that is robust to changing illumination conditions, and that is able operate in real-time. The achievement of this would allow robust human-robot interactions, and good person following behaviours, in which misclassifications of the target are less frequent despite of the existence of temporary occlusions, or periods of time in which the target is out of sight. Another good reason for focusing our work in the camera module is due to the fact that, nowadays, it is much more expensive to include in the robot a laser scanner than a conventional camera. This allows cheap robots to perform human recognition.

In the next two subsections we describe in detail the two most important parts of the camera module, the human discrimination algorithm and the online feature weighting process.

## B. Discrimination algorithm

The discrimination algorithm pursuits the differentiation of the demonstrator (target being followed by the robot) from the rest of the people moving in the same area (distractors). This algorithm runs inside the camera module (*human discrimination* block in Fig4). Basically, the discrimination algorithm will use the information of the torsos extracted from the people detected in the image, to identify whether the target is present or not. To understand the process we must realize the fact that there are two clearly different states: *initialization* stage, and *running* stage.

During an *initial stage*, when the robot is about to follow the instructor, it is assumed that the target is located in front of the robot. During this stage the system builds a model of the target by extracting the distribution of the following features from his torso: hue, lightness, saturation (colour features from the HLS colour space), local binary patterns [19] and the edges detected with the Canny method [20] (texture features). This *initial stage* is very fast and goes unnoticed for the instructor. After few instants the robot will start moving and following the target, differentiating it from the distractors (running stage).

During the *running stage*, the discrimination algorithm computes the dissimilarity between the target's model and the torsos detected in the image. First, in order to obtain the dissimilarity value, we need to compute the inverse Bhattacharyya distance between the histogram of each torso's feature, $h_f$, and the histogram of the same feature in the target's model $h_{tf}$:

$$d_f = \sqrt{1 - \sum_{i=0}^{b-1} \sqrt{h_f(i) h_{tf}(i)}} \qquad (4)$$

where $b$ is the number of bins in the histograms. Using these distances we can define the dissimilarity between each torso and the target's model as the average value of the $d_f$ for the five features being used:

$$dissimilarity = \frac{1}{n} \sum_{f=1}^{n} d_f \quad \in [0,1] \qquad (5)$$

where $n$ is the number of features, five in our case.

Finally, the discrimination algorithm will decide whether there is a torso that is similar enough to be considered the target, and whether the system should update the target's torso using the current detection. Since our dissimilarity value oscillates from 0 (very similar or equal) to 1 (a completely different torso) we have set two thresholds. The first one ($thr_1 = 0.4$) is the limit to consider a torso as an instance of the target, while the second one is a more restrictive threshold ($thr_2 = 0.2$), and it is used to decide when the target's model can be updated with the torso which is being identified as the target. This dual-threshold strategy avoids the pollution of the target's model with a false positive detection.

However, the dissimilarity measure described before (Eq. 5) is not yet robust enough to cope with real world conditions, such as strong illumination reflections, shadows, occlusions, and situations where the algorithm has to discriminate among very similar torsos. Because of this, we have designed an adaptive weighting process to dynamically enhance the differences between the target and the distractors, this process is described in the following section.

### *C. Online Feature Weighting*

This section describes the process called 'Feature weighting' in Fig. 4. This process consists on dynamically selecting the most appropriate weights for each feature to adapt to the changes in the environment, such as the illumination conditions or people's clothes, and thus enhance the discrimination of the target from the distractors. This process has been studied in the area of image retrieval with query relevance feedback. It consists on measuring the discrimination ability of each feature when differentiating between two classes. In robotics, we can define online feature weighting for human discrimination as the process of dynamically assigning high weights to those features that show a high discrimination ability between the target and the distractors. This is very useful when target and distractors show a similar distribution on some features but differ on the others. We can think of, for example, a target and several distractors wearing similar colour clothes but with different patterns. In this case it would be more useful to focus the dissimilarity on the texture while discarding the colour features.

First, to be able to enhance dissimilarity between the target and the distractors, we need to store information about the last distractors detected from the robot in a list. The distractors list is built and updated every frame according to the following rules:

1) If a torso can be classified as the target, the rest of the torsos that have been detected in the same frame will be placed on the distractor list provided that they do not overlap in the image with the torso corresponding to the target.
2) If there is no torso that can be classified as the target, those with a dissimilarity value higher than 0.5 will be put on the distractor list.
3) The list has a size limit of five torsos. When the limit is reached, the oldest torsos will be removed. This size limit of the list is set to consider only the most recently seen torsos. A larger distractor list would save torsos which will not be seen again in a short period of time, thus reducing the performance of the feature weighting process.

To assign the most suitable weights to the subset of features being used, we need a scoring function which should measure the discrimination ability of each feature at each instant. This is why we have proposed to use the Bhattacharyya distance as a score function [21]: our score is based on Eq. 4. The idea behind this score is that the best features should be the ones that minimize the distance between the last torso classified as the target and the previous target model ($d_{f,target}$, in Eq. 6) and, at the same time, it also maximizes the average distance between the distractors and the current target model ($\bar{d}_{f,distractors}$ in Eq. 6):

$$score_f = \bar{d}_{f,distractors} - d_{f,target} \tag{6}$$

Using the aforementioned function we can score the discrimination ability of each feature, and thus weight the importance of each feature in the human discrimination algorithm (Section V-B). In particular we replace Eq. 5 with a new measurement that considers the weights of the different features:

$$dissimilarity = \sum_{f=1}^{n} w_f d_f \quad \in [0, 1] \tag{7}$$

Initially, the weights are the same for all features, i.e., $w_f = \frac{1}{n}$, $\forall f = 1, ..., 5$, but as the robot proceeds moving in the environment while following the target, the weights will be updated according to Eq. 8:

$$w_f = w_f + score_f \tag{8}$$

where $w_f$ is the weight for feature $f$. Every time the weights change, it is important to re-normalize them, so that their sum is one.

## VI. Implementation and Experimental Results

We have tested the system on the Department of Electronics & Computer Science, at the University of Santiago de Compostela, Spain. The robot used in the tests is a Pioneer P3DX with a SICK-LMS200 laser and a PointGrey Chameleon CMLN-13S2C with a FUJINON FUJIFILM Vari-focal CCTV Lens (omnidirectional). On the other hand, each camera agent used either an Unibrain Fire-i camera, or an omnidirectional camera like that of the robot. The processing units where either a DELL Latitude E550 (Intel(R) Core(TM) 2 Duo P8600 @ 2.4 GHz, 4 GB RAM) or a Toshiba Satellite A100-497 (Intel(R) Core(TM) 2 Duo T5600 @ 1.83 GHz, 4 GB RAM). Regarding the software of the controllers, it was implemented using the Player(v-3.0.2)-Stage(v.4.0.0) platform for the robot, and the OpenCV 2.2 library [22] for image processing. Finally, messages were passed over an IEEE 802.11g local wireless network via UDP.

Although the intelligent camera space and the sensor data from the robot already let us record a route, we are still working on its reproduction, thus the experiments that will be presented in this section do not include route reproduction. We

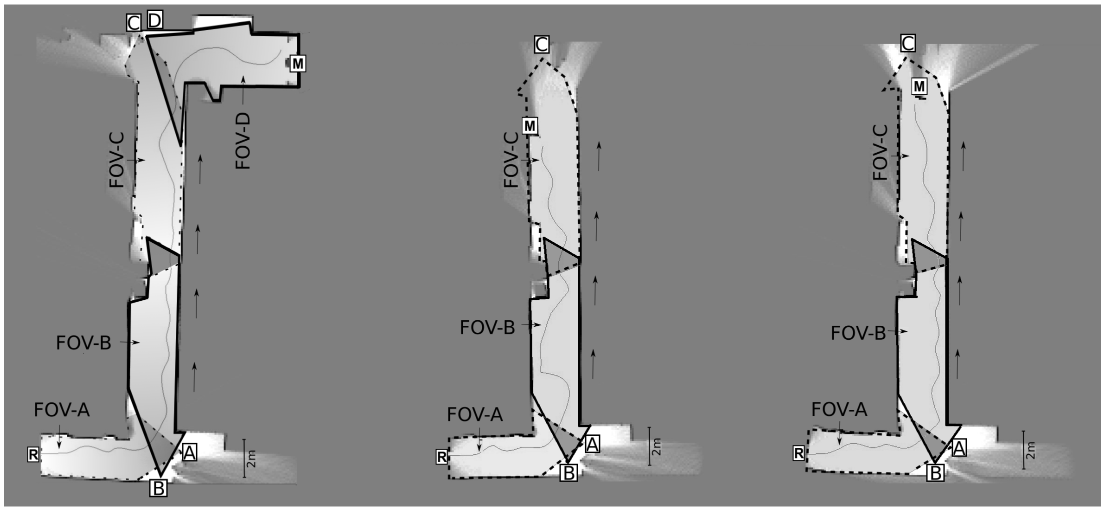

Fig. 6. Trajectories described by the robot in three tests. In the leftmost test, the robot R navigates from the FOV of camera A, passing through B, C and finally D, which triggered the call upon detection of the call event M. In the other tests, the robot navigates from the FOV of camera A, passing through B, towards C. The map and trajectory of the robot was obtained from the robot's odometry and laser readings using the PMAP SLAM library. In the figure we can also see the approximate positions of the cameras and their FOVs.

evaluated our system in two different experiments. In the first one, we tested how the cameras detect a call event, establish neighbourhood relationships among them, generate global routes, and support robot navigation to reach the call event. For simplicity, we used a colour marker in order to simulate the call event, but in the future we will include the detection of groups of people. The second experiment consisted on following a human through the department simulating a route learning process. In this experiment we focus on evaluating the performance of the human discrimination algorithm, and we also give some commentaries about the robot behaviour when following the person.

### A. Experiment I - Robot navigation to attend a call event

We deployed a multi-agent network over the Department of Electronics and Computer Science at the University of Santiago of Compostela. We have performed three different tests, represented in Figure 6.

In the first test (Fig. 6 on the left), the network consisted on a robot-agent (R), four camera-agents (A, B, C, D) and a call event (M). Camera D was the one which sighted the call event (M), started the robot call process, and triggered the route formation. Once the robot, R, received the route to follow, it started navigating through the network towards M, supported by A, B, C, and D, while avoiding the obstacles detected.

In the second and in the third tests, the network consisted on three cameras (A, B, C) instead of four. As we show in Figure 6, the performances of these tests were similar to the performance of the first one, described above.

The robot's trajectories and the maps shown in Figure 6 were obtained from the robot's odometry and laser logs using the PMAP SLAM library, compatible with Player-Stage. Nevertheless, these maps have only been used for visualization purposes, but not during the functioning of the system. We repeated these tests a few times, obtaining in all of them a satisfactory performance of the robot.

In this experiment each camera had only two neighbours, nevertheless the FOV of three or more cameras might overlap and therefore the number of neighbour cameras can be higher than two. Our system would be able to cope with this: the route that is finally selected to be followed by the robot is always the one that involves the fewest number of cameras. As part of our future work, we plan to run new experiments in wide environments where the number of cameras with overlapping FOV will be higher than two (for example in big halls).

### B. Experiment II - Person following

We have also tested the person-following behaviour through several routes in the same environment where we run the previous experiment. In each route a target walked at least 50 meters while the robot followed him, and at least one distractor was walking close to the target to evaluate the discrimination power of the online feature weighting algorithm. We have recorded one of these routes, and used it to evaluate the benefits of using our proposal over the classic approaches based on the comparison of features without weighting them. This sequence also let us test the system when the illumination conditions change significantly, altering the colour and texture properties of the torsos.

On the recorded sequence, the torsos are expected to be discriminated using texture and lightness features since both are mainly black but one has light grey drawings on it. Figure 7 shows that the results confirm what we expected: two of the features represent 80% of the total sum of weights during most of the video sequence. Analysing Figure 7 we can also notice that there is a small time interval at the beginning (from $T_0$ to $T_1$), in which there is no predominant feature and the weight values are similar to each other. This is due to the illumination conditions and pose of the target. Nevertheless this situation changes soon, achieving a small set of prominent features that increase the separability between target and distractors.

We have also selected frames where confusion could arise: the selected were all but those where the target was the only

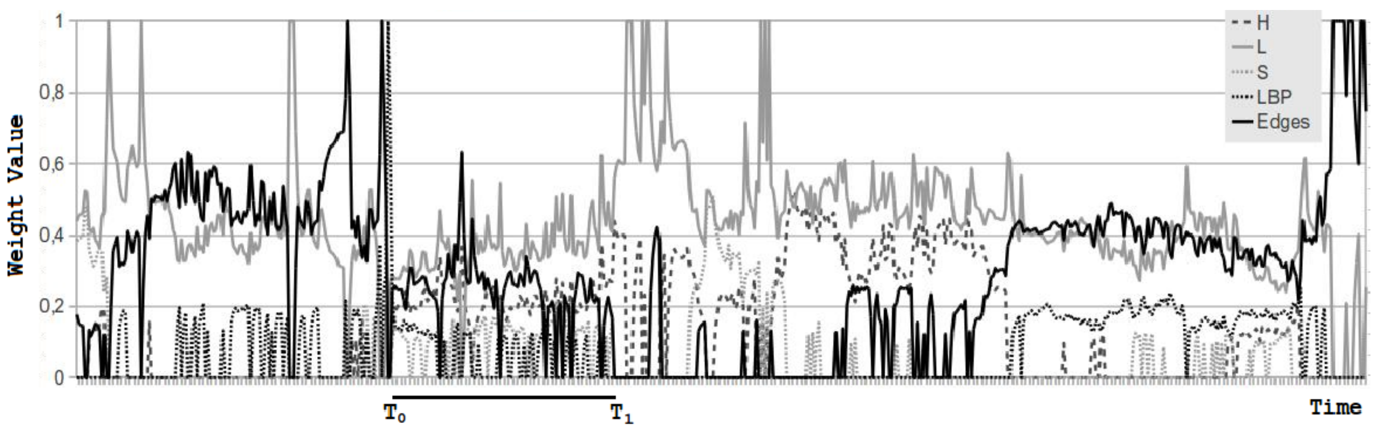

Fig. 7. Evolution of feature weights during one experiment. These weights determine the importance of each feature in the person following behaviour. According to the evolution of the weights it is noticeable that 'edges' and 'lightness' are the most important features during most of the experiment. Our algorithm is able to find the relevant features in most cases, although there are still some cases like the one between $T_0$ and $T_1$ where it is hard to find a relevant subset of features due to fast changing of the illumination conditions or the pose of the target.

TABLE I
CONFUSION MATRIX USING THE CLASSIC APPROACH

| Actual\Classif. | Target | Distractor |
|---|---|---|
| Target | 139(80.4%) | 34(19.6%) |
| Distractor | 8(3,6%) | 212(96,3%) |

detected human and thus confusion was not possible. With this selection we have built a confusion matrix out of the results observed in these frames. Table I shows the confusion matrix when using the classic approach and Table II shows the confusion matrix when using our proposed weighting of the feature space. We can see that the recognition ratio of the target increased from 80% to almost 100%. False positives decreased from eight to three reducing the number of times that the robot might follow a distractor as if he were the target.

TABLE II
CONFUSION MATRIX USING OUR APPROACH

| Actual\Classif. | Target | Distractor |
|---|---|---|
| Target | 142(99.3%) | 1(0.6%) |
| Distractor | 3(1.3%) | 217(98.6%) |

These results confirm that our system is capable of adapting to difficult conditions maximizing the dissimilarity target-distractor. We have also tested the person-following controller operating on the real robot during several 10 minute walks around the hall and different corridors of the same location as the other experiments. The robot had to follow the target when both the corridors and the premises of the building were usually crowded with students walking around. The robot's maximum speed was set to 1 m/s, thus allowing the target to walk at normal speed. The robot was able to avoid collisions with the environment thanks to a potential fields method implemented on the robot controller. The robot was able to follow the target keeping a distance that ranged from 0.4 meters to 6 meters, although the average distance between the robot and the target was 2 meters.

## VII. SUMMARY, CONCLUSIONS, AND FUTURE WORK

In this paper, we have presented a system for fast and easy deployment of guide robots in unknown environments, together with a person following behaviour, which is the basis of the route learning ability usually desirable for any advanced guide robot. The work presented here is part of a bigger project, in which we aim at developing robots which are able to participate in different social events, providing useful information to visitors. We based our design in the requirements of scalability, robustness, flexibility, and adaptability. On the one hand, we achieve fast and easy deployment by not requiring prior knowledge of the environment (e.g. metric maps), nor big expertise in order to deploy it. Moreover, we have designed it to not require any software or hardware tuning, so as to be as self-contained and automatic as possible. We have also established the basis of the route learning ability of the guide robots, by presenting an adaptive person following behaviour.

Our system consists on a distributed multi-agent network formed by two kinds of agents: intelligent cameras spread out on the environment, and autonomous robots navigating within it. The camera network senses the environment, informs robots about events happening out of their immediate surroundings (which enhances their initiative), and supports them on their duties, removing or relaxing the need of a map. We did not use any centralization or hierarchy, but biologically inspired self-organization processes, based on distribution, inter-agent independence, and emergent behaviours out of local interactions, instead of global plans. This resulted on a system highly independent of environment changes, redundant, and flexible enough to cope with a wide range of spatial distributions of the cameras.

On the other hand, the person following behaviour which we have built combines a laser based tracker, with the discrimination power of a camera. The discrimination algorithm running on the camera is inspired on image retrieval systems, which are able to adapt a set of weights for each situation that the robot might encounter. These weights enhance dissimilarity between the target being followed and the other people present in the scene, avoiding getting confused with them.

The work presented here is just the beginning of a bigger project. In future stages of our research, we will include the detection of real world events requiring robot's presence, such as groups of people interested on the services offered by the robot. Also, we will automatize the detection of overlap FOVs attending at people moving within the environment. At more advanced stages, we will explore more flexible camera arrangements, including multiple

increase the complexity of the relationships between the cameras and even remove the FOVs' overlap restriction at more advances stages. Moreover, we plan to improve the robustness of the robot navigation, by learning from people trajectories. Finally, we plan to keep improving the person following behaviour by including gesture recognition to enhance the process of route learning with a more natural interaction between the human and the robot.

## Acknowledgement

This work was supported by the research projects TIN2009-07737, INCITE08PXIB262202PR, and the grant BES-2010-040813 FPI-MICINN.

## References

[1] J. Kim, K. Lee, Y. Kim, N. Kuppuswamy, J. Jo, "Ubiquitous robot: A new paradigm for integrated services," *IEEE International Conference on Robotics and Automation*, pp. 2853-2858, 2007.

[2] J. Buhmann, W. Burgard, A. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, S. Thrun, "The mobile robot rhino," *AI Magazine*, vol. 16, no. 3, pp. 31-38, 1995.

[3] S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, et al., Minerva: "A second-generation museum tour-guide robot," *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 1999-2005, 1999.

[4] P. Trahanias, W. Burgard, A. Argyros, D. Hahnel, H. Baltzakis, P. Pfaff, C. Stachniss, "Tourbot and webfair: Web-operated mobile robots for tele-presence in populated exhibitions," *IEEE Robotics & Automation Magazine*, vol. 12, no. 2, pp. 77-89, 2005.

[5] D. Rodriguez-Losada, F. Matia, R. Galan, M. Hernando, J. Montero, J. Lucas, "Urbano, an interactive mobile tour-guide robot," *Advances in Service Robotics*, Ed. H. Seok. In-Teh, pp. 229-252, 2008.

[6] J. Lee, H. Hashimoto, "Intelligent space concept and contents,"*Advanced Robotics*, vol. 16, no. 3, pp. 265-280, 2002.

[7] J. Lee, H. Hashimoto, "Controlling mobile robots in distributed intelligent sensor network, "IEEE Transactions on Industrial Electronics," vol. 50, no. 3, pp. 890-902, 2010.

[8] J. Lee, K. Morioka, N. Ando, H. Hashimoto, "Cooperation of distributed intelligent sensors in intelligent environment," *IEEE/ASME Transactions on Mechatronics*, vol. 9, no. 3, pp. 535-543, 2004.

[9] D. Pizarro, M. Mazo, E. Santiso, M. Marron, D. Jimenez, S. Cobreces, C. Losada, "Localization of mobile robots using odometry and an external vision sensor," *Sensors*, vol. 10, no. 4, pp. 3655-3680, 2010.

[10] C. Losada, M. Mazo, S. Palazuelos, D. Pizarro, M. Marron, "Multi-camera sensor system for 3d segmentation and localization of multiple mobile robots," *Sensors*, vol. 10, no. 4, pp. 3261-3279, 2010.

[11] I. Fernandez, M. Mazo, J. Lazaro, D. Pizarro, E. Santiso, P. Martin, C. Losada, "Guidance of a mobile robot using an array of static cameras located in the environment," *Autonomous Robots*, vol. 23, no. 4, pp. 305-324, 2007.

[12] P. Steinhaus, M. Walther, B. Giesler, R. Dillmann, "3d global and mobile sensor data fusion for mobile platform navigation," *Proceedings ICRA'04. 2004 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3325-3330, 2004.

[13] P. Steinhaus, M. Strand, R. Dillmann, "Autonomous robot navigation in human-centered environments based on 3d data fusion," *EURASIP Journal on Applied Signal Processing*. vol. 2007, no. 1, pp. 224-224, 2007.

[14] M. Shiomi, T. Kanda, H. Ishiguro, N. Hagita, "Interactive humanoid robots for a science museum," *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pp. 305-312.

[15] M. Shiomi, T. Kanda, D. Glas, S. Satake, H. Ishiguro, N. Hagita, "Field trial of networked social robots in a shopping mall," *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 2846-2853.

[16] A. Sanfeliu, J. Andrade-Cetto, M. Barbosa, R. Bowden, J. Capitan, A. Corominas, A. Gilbert, J. Illingworth, L. Merino, J. Mirats, et al., "Decentralized sensor fusion for ubiquitous networking robotics in urban areas," *Sensors*, vol. 10, no. 3, pp. 2274-2314, 2010.

[17] S. Funiak, C. Guestrin, M. Paskin, R. Sukthankar, "Distributed localization of networked cameras," *Proceedings of the Fifth International Conference on Information Processing in Sensor Networks*, pp. 24-42, 2006.

[18] Dalal, N., and Triggs, B.: "Histograms of oriented gradients for human detection," *CVPR 2005, IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886-893, 2005.

[19] Ojala, T., Pietikainen, M., and Harwood, D.: "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51-59, 1996.

[20] Canny, J.: "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631-1643, 1986.

[21] Alvarez-Santos, V., Iglesias, R., Pardo, X. M., Canedo-Rodriguez, A., Regueiro, C. V.: "Online Feature Weighting for Human Discrimination in a Person Following Robot," *IWINAC 2011*, Part I, LNCS 6686, pp. 222-232, 2011.

[22] Bradski, G., and Kaehler, A., "Learning OpenCV: Computer vision with the OpenCV library," *O'Reilly Media*, 2008.