Controlling the autonomy of a reconnaissance robot

André Dalgalarrondo, Delphine Dufourd, David Filliat

Controlling the autonomy of a reconnaissance robot

André Dalgalarrondo[a], Delphine Dufourd[a] and David Filliat[a]

[a]DGA / Centre Technique d'Arcueil
16 bis avenue Prieur de la Côte d'Or, 94114 Arcueil cedex, France

ABSTRACT

In this paper, we present our research on the control of a mobile robot for indoor reconnaissance missions. Based on previous work concerning our robot control architecture HARPIC, we have developped a man machine interface and software components that allow a human operator to control a robot at different levels of autonomy.

This work aims at studying how a robot could be helpful in indoor reconnaissance and surveillance missions in hostile environment. In such missions, since a soldier faces many threats and must protect himself while looking around and holding his weapon, he cannot devote his attention to the teleoperation of the robot. Moreover, robots are not yet able to conduct complex missions in a fully autonomous mode. Thus, in a pragmatic way, we have built a software that allows dynamic swapping between control modes (manual, safeguarded and behavior-based) while automatically performing map building and localization of the robot. It also includes surveillance functions like movement detection and is designed for multirobot extensions.

We first describe the design of our agent-based robot control architecture and discuss the various ways to control and interact with a robot.The main modules and functionnalities implementing those ideas in our architecture are detailed. More precisely, we show how we combine manual controls, obstacle avoidance, wall and corridor following, way point and planned travelling. Some experiments on a Pioneer robot equipped with various sensors are presented. Finally, we suggest some promising directions for the development of robots and user interfaces for hostile environment and discuss our planned future improvements.

Keywords: Ground robotics, autonomy, human computer interface, indoor navigation, reconnaissance robot.

1. INTRODUCTION

In November 2003, the French defense procurement agency (Délégation Générale pour l'Armement) launched a prospective program called "PEA Mini-RoC" dedicated to small unmanned ground systems. This program focuses on platform development, teleoperation and mission modules. Part of this program aims at developing autonomous functions for robot reconnaissance in urban terrain. In this context, the Centre Technique d'Arcueil (CTA) of the Délégation Générale pour l'Armement (DGA) is currently conducting studies to demonstrate the potentialities of advanced control strategies and to get a first feedback from operational forces about the use of small robotic platforms during military operations in urban terrain. Our goal is not to elaborate operational systems but to investigate several solutions[1] on experimental platforms in order to suggest requirements and to be able to build specifications for future systems. Our research focuses on robot localization,[2] autonomous map building[3,4] and robot control architecture.[5]

1.1. Overview of existing paradigms

Many paradigms have been developed for human robot interaction. They allow various levels of interaction with different levels of autonomy and dependance. In this paragraph we attempt to list briefly the main paradigms from the less autonomous (teleoperation) to the most advanced (mixed-initiative) where humans and robots can cooperate to determine their goals and strategies.

Further author information:
A.D.: E-mail: andre.dalgalarrondo@etca.fr, phone: +33 (0)1 42 31 96 59
D.D.: E-mail: delphine.dufourd@etca.fr, phone: +33 (0)1 42 31 97 07
D.F.: E-mail: david.filliat@etca.fr, phone: +33 (0)1 42 31 97 19

Teleoperation is the most basic and mature mode. In this mode the operator has full control of the robot and must shoulder total responsibility for mission safety and success. This mode is suitable in complicated or unexpected situations that no algorithm can deal with. In return this mode often means a heavy workload for the teleoperator who needs to focus his attention on the task. With *Supervisory control*[6] the operator (called the supervisor) orders a subordonate (the robot) to achieve a predefined plan. In this paradigm, the robot is merely a tool executing tasks under operator monitoring. This interaction mode can adapt to low level bandwith and high level control but needs constant vigilance from the operator who must react in case of failures (to perform manual mission re-planning for example). This mode is only suitable for static environments where planning is really effective. *Behavior-based teleoperation*[7] replaces fine-grained control by robot behaviors which are locally autonomous. In comparison to supervisory control, this paradigm brings safety and provides the robot with the opportunity to react to its own environment perception (.e.g. to avoid obstacles). Thus, it allows the operator to be more negligent. *Adjustable autonomy*[8] refers to the adjustment of the level of autononomy which has been initiated by the operator, by another system or by the system himself and while the system operates. The goal of this paradigm is generally to increase the negligence time allowed to the operator while maintaining the robot to an acceptable safety and effectiveness level. In *Traded control*[9] the operator controls the robot during one part of the task and the robot behaves autonomously during the other part of this task. This paradigm may lead to an equivalent of the kidnapped robot problem. While the robot is controlled by the operator, it can loose its situation knowledge and face difficulties when coming back to autonomous control. In *Shared control*,[10] part of the robot functions are autonomous and the remaining are controlled by the operator. It is important to notice that this mode requires constant attention from the operator. If the robot is only in charge of some safety mechanisms, this paradigm is equivalent to the *safeguarded teleoperation*. *Mixed-initiative*[11] is characterized by a continuous dialogue between the operator and the robot where both of them share decision and responsibility. *Collaborative control*[12] may be described as a restrictive implementation of mixed-initiative. In this approach, the human robot dialogue is mainly reduced to a predefined set of questions and answers.

All these paradigms may be considered as subsets of the human robot teaming domain with important overlapping. Therefore, many robotic experimentations use a mix of these paradigms.

1.2. Application to military reconnaissance robot

Military unmanned ground robots for scout or search and rescue missions have to deal with ill-known and hostile environments. Thus, in urban warfare, they face many uncertainties and intermix with friends, foes and bystanders. Technology is not mature enough to produce autonomous robots that can handle these situations on their own. Therefore robots should be used as force multipliers and risk reducer in land operations. It seems that today's best solutions have to rely on a good collaboration between humans and robots e.g. when robots act as autonomously as possible but remain under human supervision.

Here is an example of a scenario involving collaboration between soldier teams and robots. Robots could be used for reconnaissance in a building with many floors. The reconnaissance of each floor is done by a team of soldiers and robots. The robots may be carried on each floor by one or two soldiers. Another team is outside the building, surveys the environment and supervises the whole mission. At each floor, one or more robot are ahead from the soldiers and explore the floor. An operator who is responsible for robot supervision is protected by his team members so that he does not have to watch too much around him and to hold his weapon in case of imminent danger. When a team progresses into the building, the robot control is exchanged between soldiers so that the previous operator can move and follow the maneuvers safely. The coordination within the team is made by direct voice, radio or by signs. The images and data from the robots can be viewed by each team member in order to share information. Each robot can perform moving objects detection, heat and sound detection, map building and localization, navigation behaviors and give access to its data. The team which remains outside the building can also have access to this information so as to construct a global picture of the mission and to watch its evolution. In particular, it can monitor the map built at each floor by the robots and regulate the progression of the teams. When a resistance is encountered in a floor, the outdoor team can plan an action from the outside and through the appropriate windows that they have located on the same map.

Basic capacities that have been shown on many robots include obstacle detection and avoidance, waypoint and goal oriented navigation, detection of people, detection of threats, information retrieval (image, sound)

localization and map building, exploration, coordination with other robots. Most of them can be implemented on a real robot with a good reliability in reasonably difficult environnement but it is not enough to fullfill military requirements. As a partner or as an extent of a warfighter, a robot must nether increase the risk for the team – which stands for "surprise effect" cancelling and trap triggering for instance – nor impose an important workload to the human supervisor.

We think that in short term robots will not be able to handle some uncertain situations and that the most challenging task is to build a software organization which provides a pertinent and adaptative balance between robot autonomy and human control. In such teams, humans are superior in perception (especially in environment understanding), in knowledge management and in decision making. Robots can be better than humans in quantitative low-level perception, in precise localization and in displacement in confined and cluttered space. Above all, robots can stand dull, dirty and dangerous jobs. So, for a good teaming, it is desirable that robots and humans share their capacities along a two-way dialogue. This is the approximate definition of the mixed-initiative control mode but, given the maturity of today's robots and the potential danger for soldiers, we do not think that mixed-initiative could be the solution for now. It seems that adjustable autonomy, with a wide range of control modes – from basic teleoperation to even limited mixed-initiative – is the most pragmatic and efficient way.

However, independantly from the control mode, the design of an easy to handle and ergonomic operator control unit remains a major issue. A review of the research in human robot interaction is beyond the scope of this paper and of our work but we can notice that most recent work have adopted a graphical interface running on a PDA. In our development, we follow this trend and try to build a PDA-sized interface allowing a seamless and natural transition between various levels of autonomy for an indoor reconnaissance robot. This work is presented in the next paragraph.

2. PRESENTATION OF OUR WORK

Our work is based on a multiagent robot control architecture named HARPIC which was developed in[13] and which is briefly described in the next paragraph. Then we present the specific agents and the organization which allows the various control modes.

2.1. General description of HARPIC

HARPIC is an hybrid architecture (cf. figure 1) which consists in four blocks organized around a fifth: *perception processes*, an *attention manager*, a *behavior selector* and *action processes*. The core of the architecture (the fifth block) relies on *representations*.

Sensors yield data to perception processes which create representations of the environment. Representations are instances of specialized perception models. For instance, for a visual wall-following behavior, the representation can be restricted to the coordinates of the edge detected in the image, that stands for the wall to follow. To every representation are attached the references to the process which created it: date of creation and various data related to the sensor (position, focus...). The representations are stored in a table with fixed constant length, so that a round-robin mechanism keeps a given memory depth. Thus, representations are snapshots of specific landmarks in the robot's environment, the spatial and the temporal localization of which are known.

The perception processes are activated or inhibited by the attention manager and receive information on the current behavior.[14] This information is used to foresee and check the consistency of the representation. The attention manager has three main functions: it updates representations (on a periodical or on an exceptional basis), it supervises the environment (detection of new events) and the algorithms (prediction/feedback control), and it guarantees an efficient use of the computing resources. The action selection module chooses the robot's behavior depending on the predefined goal(s), the current action, the representations and their estimated reliability. Finally, the behaviors control the robot's actuators in closed loop with the associated perception processes.

The key ideas of this architecture are:

• *The use of sensorimotor behaviors linking perceptions and low-level actions both internally and externally*: the internal coupling allows to compare a prediction of the next perception (estimated from the previous perception

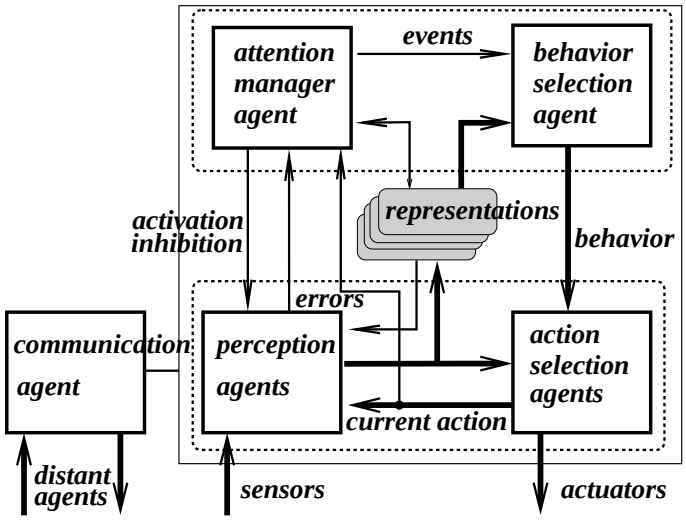

Figure 1. Functional diagram of the HARPIC architecture.

and the current control) with the perception obtained after application of the control, in order to decide whether the current behavior runs normally or should be changed; the external coupling is the classic control loop between perception and action.

• *Use of perception processes* with the aim of creating local situated representations of the environment. No global model of the environment is used; however more global and higher level representations can be built from the instantaneous local representations.

• *Quantitative assessment of every representation*: every algorithm is associated with evaluation metrics which assign to every constructed representation a numerical value expressing the confidence which can be given to it. We regard this assessment as important feature[15] since any processing algorithm has a limited domain of validity and its internal parameters are best suited to some situations only. There is no perfect algorithm that always yields "good" results.

• *Use of an attention manager*: it supervises the execution of the perception processing algorithms independently from the current actions. It takes into account the processing time needed for each perception process, as well as the cost in terms of computational resources. It also looks for new events due to the dynamics of the environment, which may signify a new danger or opportunities leading to a change of behavior. It may also trigger processes in order to check whether the sensors operate nominally and it can receive error signals coming from current perception processes. In practice, for instance with a vision sensor, the attention will focus on the illumination conditions, on the consistency between the movement of the robot and the temporal consistency of the representations, and on error signals sent by perception processes. With this information it becomes possible to invalidate representations due to malfunctioning sensors or misused processes.

• *The behavior selection module* chooses the sensorimotor behaviors to be activated or inhibited depending on the predefined goal, the available representations and the events issued from the attention manager. This module is the highest level of the architecture. It should be noted that the quantitative assessment of the representations plays a key role in the decision process of the behavior selection.

On the one hand, a representation might be more or less adapted to the current situation, depending for instance on the sensor used or on the conditions of the perception acquisition. For instance, a day camera used during night operating conditions will yield representations to which a lower confidence should be assigned a priori. The same also holds for instance for a detector relying on translational invariance while the robot has such a motion that this assumption is incorrect.

On the other hand, some representations might be more interesting for some behaviors or might provide an improved help to choose between several behaviors (e.g. a wall-following behavior needs information on contours rather than on velocity vectors, while a tracking behavior has the opposite needs). Therefore every behavior also weighs each representation depending on its direct usability, and this weight is combined with the intrinsic assessment of the representation.

• *The action selection module* regroups the lower-level controllers operating on the actuators. It uses valid representations in order to compute the control laws.

2.2. HARPIC implementation

Fundamental capacities of our architecture encompass modularity, encapsulation, scalability and parallel execution. To fulfill these requirements, we decided to use a multi-agent formalism that fits naturally our need for encapsulation in independent, asynchronous and heterogeneous modules. The communication between agents is realized by messages. Object oriented language are therefore absolutely suited for programming agents: we chose C++. We use POSIX threads to obtain parallelism: each agent is represented by a thread in the overall process. For us, multi-agent techniques is an interesting formalism and though our architecture could be implemented without them it led to a very convenient and scalable framework.

All the agents have a common structure inherited from a basic agent and are then specialized. The basic agent can communicate by sending messages, has a mailbox where it can received messages and runs its own process independantly of orther agents. Initially, all present agents have to register to a special agent called the administrator that record all information about agents (name, type, representation storage address...). All these data can be consulted by any agent. Then, when an agent is looking for another one for a specific job to do, it can access to it and to its results. It is for example what is happening when an action agent has to use a representation coming from a perception agent.

Perception and action agents follow the global scheme. The action agent is activated by a specific request coming from the behavior selection agent. The selection orders him to work with a perception agent by sending its reference. The action agent sends in turn a request to the proper perception agent. Perception agents are passive, they only run upon request, perform a one shot execution and then wait for a new message. Furthermore, a perception agent can activate another agent and build a more specific representation using its complementary data. Many action and perception agents run at the same time but most are waiting for messages. Only one behavior (composed of a perception agent and an action agent) is active at the same time. Within a behavior, it is up to the action agent to analyze representations coming from the perception agent and to establish the correct control orders for the platform.

The attention agent supervises the perception agents. It has a look-up table where it can find the types of perception agents it has to activate depending on the current behavior. It is also in charge of checking the perception results and of declaring new events to the behavior selection agent when necessary. The advantage of this organization is detailed in previous papers.[5, 16]

The selection agent has to select and activate the behavior suited for the robot mission. This agent may be totally autonomous or constitute the process that runs the human computer interface. In this work, it is the second case and this agent is detailled in paragraph 2.4.

We use two specific agents to link our architecture to hardware. The first one is an interface between the software architecture and the real robot. This agent is awaiting a message from an action agent before translating the instructions into comprehensible orders for the robot. Changing the real robot require the use of a specific agent but no change in the overall architecture. The second agent acquires images from the grabber card at a regular rate and stores them in computer memory which can be consulted by perception agents.

Finally, we use a specific agent for IP communication with distant agents or other architectures. This agent has two running loops: an inner loop in which it can intercept messages from other agents belonging to the same architecture, and an external loop to get data or requests from distant architectures. This agent surpervises the (dis)appearance of distant agents or architectures. It allows the splitting of an architecture on many computers or the communication between several architectures. For example, this agent is useful when agents are distributed between the robot onboard computer and the operator control unit.

Figure 2. Interface for laser-based (left), image-based (center) teleoperation and goal navigation (right).

2.3. Agents for SLAM and path planning

Map building is performed by a perception agent that takes laser sensor data and odometry data as input and outputs a representation which contains a map of the environment made of registered laser scans. The map building technique is based on the simple and efficient mapping algorithm[17] that makes use of histogram correlation for scan matching. This algorithm can handle fairly large environments and is quite robust to large cycle closure. It proves to be very robust as long as some straight walls are present in the environment but may fail in extremely cluttered area. This agent is executed whenever new laser data are available (e.g. 5 Hz), but it adds data to the map only when the robot has moved at least one meter since the last map update.

Localization is performed by a perception agent that takes odometry, laser data and the map representation as input and outputs a representation containing the current position of the robot. In its current implementation, this agent takes the position periodically estimated by the mapping algorithm and interpolates between these positions using odometry data to provide anytime position of the robot. This agent is executed upon request by any other agent that has to use the robot position (e.g. mapping, planning, human computer interface...). Despite the fact that it is not useful in the current implementation, we have chosen to separate the localization agent from the map-building agent so as to be easily able to use either pre-recorded maps or other localization methods for comparison and evaluation purposes. For example, we are currently implementing a particle filter approach to localization[18] in order to evaluate the potential gain in robustness brought by this method.

Finally, path planning is carried out by an action agent that takes the map and position representations as inputs and outputs motor commands that drive the robot toward the current goal. This agent first converts the map into an occupancy grid and using a value iteration algorithm it computes a potential that gives for every cell of the grid the length of the shortest path from this cell to the goal. The robot movements are then derived using gradient descent on this potential from the current robot position. The path planning agent is used in the *Navigation* control mode of the operator control unit (described below).

2.4. HCI agent

In this implementation, our human computer interface is a graphical interface managed by the behavior selection agent of HARPIC. It is designed to use a small touch-screen of 320x240 pixels such as the one that equipped most personal digital assistant (PDA). With this interface, the user has to select a screen coresponding to one of

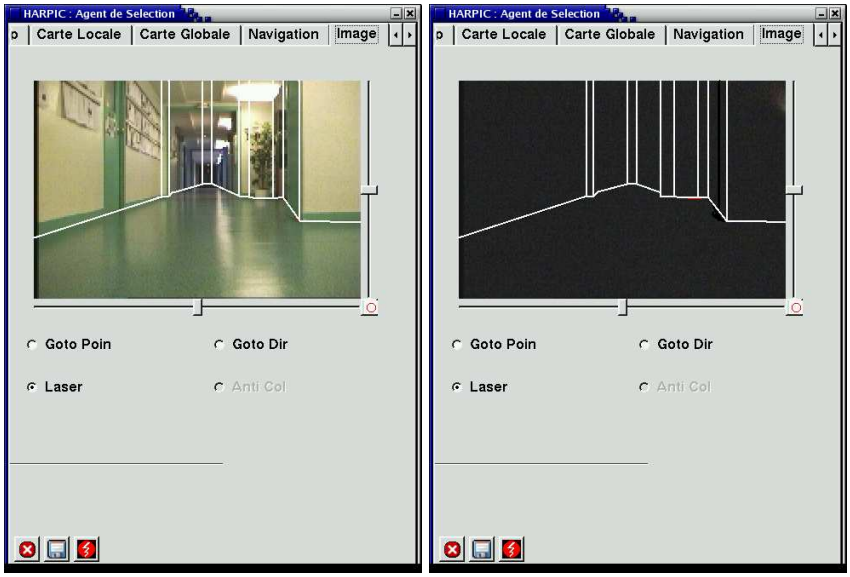

Figure 3. Image screen in normal (left) and in low-light condition (right) with overlaid polygonal view.

the control mode he wants to activate or a screen showing the environement measures and representations built by the robot. These screens are described below.

The *Teleop* screen corresponds to a teleoperation mode where the operator controls the translational and the rotational speed of the robot (see figure 2 (left)) by defining on the screen the end of the speed vector. The free space measured by the laserscanner can be superimposed on the screen and it appears in white color. The operator can also activate anti-collision and obstacle avoidance functions. Messages announcing obstacles are also displayed. This screen allows full teleoperation of the robot displacement (with or without seeing it thanks to the laser free space representation) as well as safegarded teleoperation. As a result the operator has full control on the robot in order to trade with precise movement in cluttered space or with autonomous mobility failure. In countrepart, he must keep defining the robot speed vector otherwise it stops and waits.

The *Image* screen allows to control the robot by pointing at a location or defining a direction within the image acquired by the onboard robot camera. As the previous mode, this one enables full control or safeguarded teleoperation for the robot displacement. Two sliders operate the pan and tilt unit of the camera. When the *GoTo XY* function is enabled, the location selected by the operator in the image is translated into a robot displacement vector by projection on the ground plane with respect to the camera calibration and orientation angles. The *Direction* function moves the robot when the operator defines the end of the speed vector on the screen. The selectable *Laser* function draws a polygonal view of the free space in front of the robot (in an augmented reality way) which is built from the projection of the laser range data in the image. This *tron-like* representation allows to control the robot in the image whenever there is no sufficient light for the camera. Incidentally, this function provides a visual checking of the good correspondance between the laser data and the camera data. Figure 3 illustrates the effect of this function. If the *GoTo XY* or *Direction* functions are not enabled and if the robot is in any autonomous mode, this screen can be used by a operator to supervise the robot action by viewing the images of the onboard camera. However, it can stop the robot in case of emergency.

The *Navigation* screen shows a map of the area already explored by the robot. In this control mode, the operator has to point out a goal location in the map to trigger an autonomous displacement of the robot until that goal. The location can be changed dynamically (whenever the previous goal has not been reached). The planification process is immediate (about some seconds). When new areas are discovered, the map is automatically updated. As shown in figure 2 (right), the map is an occupancy grid where bright areas correspond to location which have been observed as free more often than darkest area. Three sliders can translate or zoom

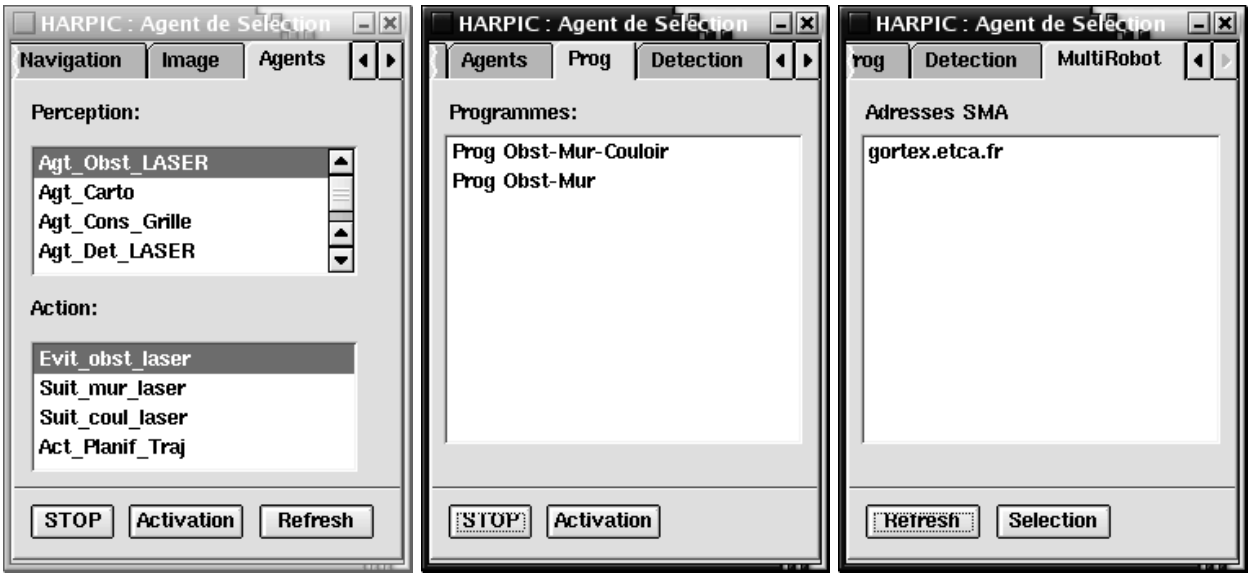

Figure 4. Interface for agent (left), program (center) and robot (right) selection.

the displayed map. This is an autonomous control mode where the operator can select a goal and forget the robot.

The *Agents* screen allows the composition and the activation of behaviors by selecting an appropriate pair of perception and action agents (see fig. 4 (left)). For example, it allows to execute behaviors like obstacle avoidance, wall following or corridor following with different perception or action agents (each one corresponding to a specific sensor or algorithm) for a same behavior. For example, a wall following behavior can result from a perception agent using the camera, from another one using the laserscanner and from various algorithms. This control mode correspond to the behavior-based teleoperation paradigm. However this screen has been mainly designed for expert users and development purpose. It lacks simplicity but it will be easily reduced to a small number of buttons when the most effective behaviors set will be determined.

The *Prog* screen corresponds to predefined sequences of behaviors. For example, it enables the robot to alternate obstacle avoidance and wall or corridor following when respectively an obstacle, a wall or a corridor appears in the robot trajectory (see fig. 4 (center)). This example is a kind of sophisticated wander mode. More generally, this mode allows automous tasks that could be described as a sequence of behavior like exploration or surveillance where observation and navigation behaviors are combined. The list of these sequences can be easily augmented to adapt the robot to a particular mission. In this control mode, the robot is autonomous and the operator workload could be null.

A *MultiRobot* screen allows the operator to select the robot which will be controlled by his control unit. Indeed, our interface and sofware architecture is designed to adress more than one robot. In a platoon with many robots this capacity may give way to the sharing of each robot data or representation and to the exchange of robot control between soldiers.

The *Local Map* and *Global Map* screens show the results of the SLAM agents described in 2.3 (see fig. 5 (left and center)). The first one is a view of the free zone determined on each laser scan. The second one displays the global map of the area explored by the robot and its trajectory. The circles on the trajectory figure the location where the SLAM algorithm has added new data. The current position of the robot is also shown on the map. As on some others screens, sliders allow the operator to translate and to zoom the display. These screens may be used when the robot is in any autonomous mode to supervise its movements.

The *Detection* screen displays moving objects trajectories in the map built by the robot (see fig. 5 (right)). This screen stands for surveillance purpose. The algorithm used is based on movement detection on laser range

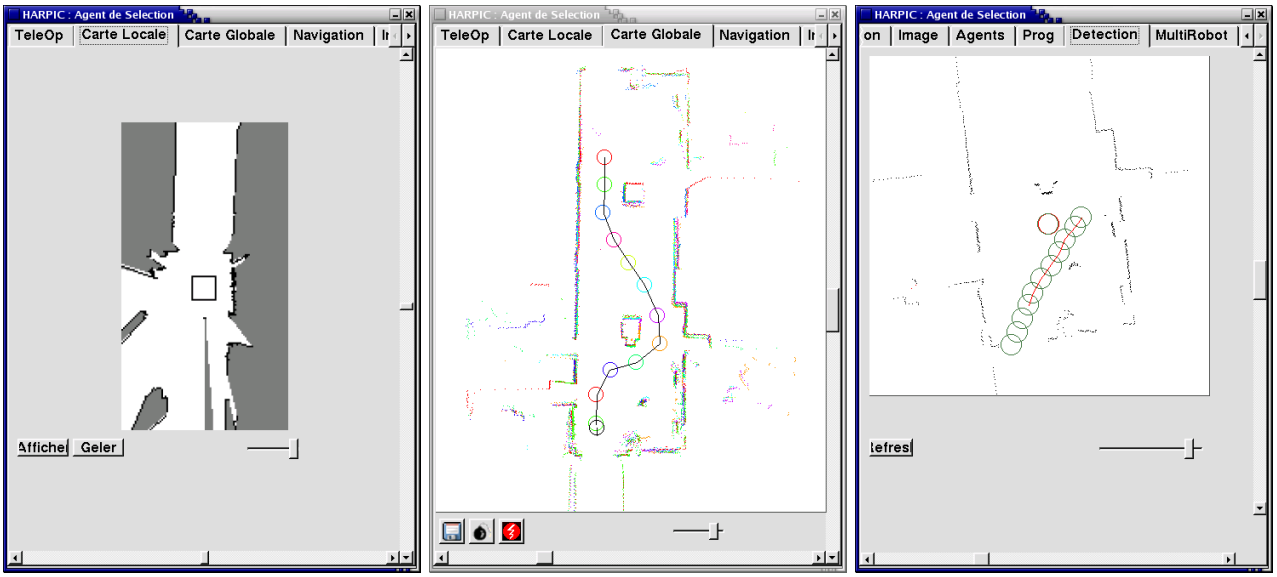

Figure 5. Interface for local map (left), global map (center) and moving object detection (right).

data and Kalman filtering. As it refers to the map built and updated by the robot, the detection and tracking of moving objects can be done while the robot is moving. This screen shows that this interface is not limited to displacement control of the robot but can be extended to many surveillance tasks.

The transition between any autonomous or teleoperation screens causes the ending of the current action or behavior. These transitions have been designed to appear natural to the operator. However, when one of these modes is actived, it is still possible to use the screens that display robot sensors data or representations without disactivating them. This feature is valid for the operator control unit but also for other control units. Thus, in a soldier team images and representations from a given robot can be viewed by a team member that is not responsible for the robot operation.

2.5. Experimentation

The robot we used both in indoor and outdoor experiments is a Pioneer 2AT from ActivMedia equipped with sonar range sensors, color camera (with motorized lens and pan-tilt unit) and an IBEO laser range sensor. On board processing is done by a rugged Getac laptop running Linux and equipped with IEEE802.11 wireless link, frame grabber card and Arcnet card for connection to the laser (cf. figure 6). We also use another laptop with the same wireless link that plays the role of the operator control unit. The agents of the robot control architecture are distributed on the two laptops. We did not use any specialized hardware or real-time operating system.

Several experiments have been conducted in the rooms and corridors of our building and have yielded good results. In such an environment with long linear walls or corridors, the automous displacement of the robot using the implemented behaviors is effective. However, in this particular space, a few limitations for the SLAM process have been identified. They mainly come from the laser measurement when the ground plane hypothesis is not valid and in the presence of glass surfaces.

The largest space we have experimented so far was the exhibition hall of the JNRR'03 conference.[19] Figure 7 shows the global map and the robot trajectory during this demonstration. It took place in the machine-tool hall of the Institut Français de Mécanique Avancée (IFMA) and in the presence of many visitors. As could be seen on figure 7, these moving objects introduced some sparse and isolated edges on the map but did not disturb the map building process. The robot travelled an area about 60×60 m large with loops and abrupt heading changes. The robot displacement was mainly done in the safeguarded teleoperation mode because the building lacked main structures and direction for the robot to follow and because of the presence of people.

Figure 6. The experimental platform.

These experiments have revealed some missing functions in our interface (e.g. "return to starting point" behavior, manual map management such as the limitation of the map space, etc) but no deep change requirement in the software architecture have been discovered.

2.6. Future work

Our human computer interface runs on a PC laptop with Linux and Qt C++ toolkit for the graphic interface. It is currently being integrated on a PDA with Linux and Qtopia environment. New functions and behaviors will appear within the next months. We are working on behaviors for people following that are based on image and laser data processing. They will allow an operator to make a robot follow him or another team member. The integration of a catadioptric camera to cope with the narrow field of view of the actual onboard camera is also on the run. Next we plan to develop autonomous behaviors for exploration, go-back-home and for assistance in narrow space crossing like doors. In the meantime, we keep improving existing behaviors so as to make them as robust as possible. Development of more multirobot capacity, interaction and cooperation are also planned as a second step. Our work focuses on function development and integration and not on ergonomy and usability of the interface. We think that such studies are necessary but will have to be conducted after the definition of the functions of the interface and their transition mechanisms.

3. SUMMARY

This article describes our work concerning the development of an effective software to control a reconnaissance robot. We have created a robot control architecture based on a multiagent paradigm that allows various levels of autonomy and interaction between an operator and the robot. According to the state of the art in robot autonomy in hostile environment we think that *adjustable autonomy* is the most pertinent choice for the next generation of military robots. Our work shows that many control modes with seamless transitions can be fairly integrated in a quite simple operator control unit. It has also confirmed the good behavior of our software architecture HARPIC and the great advantage of the flexible multiagent approach when adding many functions. Moreover this kind of interface is a good mean to experiment, evaluate or demonstrate autonomous robot behaviors and human robot interaction. We hope that it will soon enable us to gather more military requirements and to improve the specification of future systems.

Figure 7. Map and robot trajectory generated during a JNRR'03 demonstration in the IFMA hall. Each circle on the robot trajectory represents the diameter of the robot which is about 50 cm.

ACKNOWLEDGEMENTS

This work was entirely performed at the Geomatics-Imagery-Perception department of the Centre Technique d'Arcueil within the DGA. We would like to thank S. Moudere, E. Sellami, N. Sinegre, F. Souvestre, D. Lucas, R. Cornet and G. Calba for their contributions to the software development of this work.

REFERENCES

1. D. Luzeaux, A. Dalgalarrondo, and D. Dufourd, "Autonomous small robots for military applications," in *UVS Tech 2001*, (Belgian Royal Military Academy, Brussels, Belgium), 6-7 december 2001.
2. D. Filliat and J.-A. Meyer, "Map-based navigation in mobile robots: I. a review of localization strategies," *Cognitive Systems Research* **4 (4)**, pp. 243–282, december 2003.
3. D. Dufourd, "Autonomous construction of indoor maps with a mobile robot," in *SPIE* 15^{th} *Annual Symposium, AEROSENSE'01, Unmanned Ground Technology III*, (Orlando, FL), april 2001.
4. D. Filliat and J.-A. Meyer, "Map-based navigation in mobile robots: Ii. a review of map-learning and path-planning strategies," *Cognitive Systems Research* **4 (4)**, pp. 283–317, december 2003.
5. D. Luzeaux and A. Dalgalarrondo, *Hybrid architecture based on representations, perception and intelligent control*, ch. Studies in Fuziness and Soft Computing: Recent Advances in Intelligent Paradigms and Applications. ISBN-3-7908-1538-1, Physica Verlag, Heildelberg, 2003.
6. P. Backes, K. Tso, and G. Tharp, "The web interface for telescience," in *IEEE International Conference on Robotics and Automation, ICRA'97*, (Albuquerque, NM), 1997.
7. M. Stein, "Behavior-based control for time-delayed teleoperation," Tech. Rep. 378, GRASP Laboratory, University of Pennsylvania, 1994.
8. G. Dorais, R. Bonasso, D. Kortenkamp, B. Pell, and D. Schreckenghost, "Adjustable autonomy for human-centered autonomous systems," in 6^{th} *International Joint Conference on Artificial Intelligence (IJCAI), Workshop on Adjustable Autonomy System*, 1999.
9. D. Kortenkamp, R. Bonasso, D. Ryan, and D. Schreckenghost, "Traded control with autonomous robot as mixed initiative interaction," in 14^{th} *National Conference on Artificial Intelligence*, (Rhode Island, USA), july 1997.
10. T. Röfer and A. Lankenau, "Ensuring safe obstacle avoidance in a shared-control system," in *Seventh International Conference on Emergent Technologies and Factory Automation, EFTA'99*, (Barcelonna, Spain), 1999.
11. G. Ferguson, J. Allen, and B. Miller, "Trains-95: Towards a mixed-initiative planning assistant," in *Third International Conference on AI Planning Systems, AIPS-96*, 1996.
12. T. Fong, C. Thorpe, and C. Baur, "Collaborative control: a robot-centered model for vehicle teleoperation," in *AAAI Spring Symposium on Agents with Ajustable Autonomy, Technical report SS-99-06*, (Memlo Park), 1999.
13. A. Dalgalarrondo, *Intégration de la fonction perception dans une architecture de contrôle de robot mobile autonome*. PhD thesis, Université de Paris-Sud, Centre d'Orsay, janvier 2001.
14. A. Dalgalarrondo and D. Luzeaux, "Dynamic selection of perception process on an autonomous robot," in *SPIE* 13^{th} *Annual Symposium, AEROSENSE'99, Vol. 3691*, (Orlando, FL), april 1999.
15. A. Dalgalarrondo and D. Luzeaux, *Assessment and applications assessment of image processing algorithms as the keystone of autonomous robot control architectures*, ch. Imaging and Vision Systems, Advances in Computation: Theory and Practice. NOVA Science Books, New York, 2001.
16. A. Dalgalarrondo and D. Luzeaux, "Introducing attention in a behavior-based robot control architecture," in *European Control Conference, ECC'99*, (Karlsruhe, Germany), september 1999.
17. T. Röfer, "Using histogram correlation to create consistent laser scan maps," in *IEEE International Conference on Robotics Systems (IROS-2002)*, (EPFL, Lausanne, Switzerland), 2002.
18. S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence* **128 (1-2)**, pp. 99–141, may 2001.
19. "Quatrièmes Journées Nationales de Recherche en Robotique (JNRR'03)," (Clermont-Ferrand, France, http://lasmea.univ-bpclermont.fr/jnrr03/), 8-10 octobre 2003.