10

A User Multi-robot System Interaction Paradigm for a Multi-robot Mission Editor

Saidi Francois and Pradel Gilbert
IBISC Laboratory, FRE CNRS 2873, Université d'Evry-Val-d'Essonne France

1. Introduction

Many applications such as space and underwater exploration, operations in dangerous environment, service robotics, military applications, etc. can call upon multi-robot systems. These systems, although far from achievement, can carry out difficult or even tasks impossible to achieve by a single robot. A team of robots provides a certain redundancy, contributes to the achievement of a task in a collaborative way and should be able to go beyond what could be done by a single robot.
According to (Parker, 2000) and (Arai & al., 2002), works in collective mobile robotics can be classified in three categories:

- Reconfigurable robots systems also called "Cellular Robots Systems". A cellular robot is an auto-organized robot-like system composed of a large number of units called cells. This idea is inspired by the organization of a living system. Various fields were studied in this domain, in particular the swarm intelligence (Bonabeau & Theraukaz, 2000) (Beni & Hackwood, 1992) on the "cyclic swarms". One will also note the work of Fukuda (Fukada & Nakagawa, 1987) on the system CEBOT.
- Trajectory planning: one can quote the works (Premvuti & Yuta, 1990) on the control of the aircraft traffic, of (Arai & al., 1989) and of (Wang, 1989) on the movement of groups of robots in formation.
- Architectures for multi-robot co-operation among which the system ACTRESS (Asama & al., 1989) (Sellem & Dalgalarrondo 1999). ACTRESS deals with maintenance in a known and structured environment. It resolves conflicts between robots and allocates tasks by creating staffs of robots.

In this paper, our interest is mostly focused on the human-system interactions. Studies on human-system interactions show that, to use efficiently the system, the user has to "accept" it. This acceptance implies that two concepts are to be taken into account: usability and appropriation (section 0 and 0). The user also needs to be permanently informed on the activity of the system, on the individual activities of its entities, on the state of the different components to compare the mental representation of the mission progress he/she has at the beginning of the mission and what is really executed by the system. This is peculiarly true when a user (e.g. a handicapped person, a person with specific needs) requests services to a multi robot system.

Source: Recent Advances in Multi-Robot Systems, Book edited by: Aleksandar Lazinica, ISBN 978-3-902613-24-0, pp. 326, May 2008, I-Tech Education and Publishing, Vienna, Austria

In the case of the teleoperation of a system, the user must compensate for sensory impoverishment, replace the sensorimotor mechanisms usually used, and learn new strategies to control the distant system. Initially, researchers sought to make the system the most autonomous possible. They realized that its complexification, in the design and in the realization, led to great difficulties of utilisation by the increase of what one could call the distance between the orders necessary to the machine and the natural reactions of the user. A second approach tends to design machines which interact with the user by studying the distribution of the tasks between the human and the system and how this user uses the system to conceive/improve this one. This use of the system by the operator is typical of the way in which he/she adapted to the machine. This is much more difficult when the system is a multi-robot system. These features led us to consider the system design as a participative system. These participative aspects will involve:

- the Human-System interface (HSI),
- the service request seen as a set of remote services carried out either in an individual or in a collective way by the robots,
- the user intervention in the construction of the solution,
- a way to act directly on the robots or on the mission scene in the event of modification, breakdown or execution failure,
- an always available information on the system state.

This paper presents an original approach of the user/multi-robot system interaction. After a review of the state of the art in collective robotics (section 0), we will introduce our participative model for human system interaction (section 0). This system design will call upon several levels of abstraction, the first level, the semantical and syntactical verificator, is discussed in section 0, the second level, the geometrical verificator, is analysed in section 0. This paper ends with a conclusion and some prospects.

2. Mobile collective robotics

The works of (Balch & Parker, 2002);(Schultz & Parker, 2002);(Parker & al., 2002) define 6 directions of study in multi-robot systems:

1. biologically inspired systems,
2. systems which study the communications,
3. systems which are interested in architectures, tasks allocation and control,
4. systems for objects transport and handling,
5. systems for displacements coordination,
6. systems dealing with the design of reconfigurable robots.

2.1 Biologically inspired systems

Most of multi-robot systems of biological inspiration follow upon the "behaviour-based" paradigm (behavioural robotics) of Brooks (Brooks, 1986). Researchers (Drogoul & Ferber, 1993) were interested in modelling insects or animals societies (ants, bees, birds, fish...) to reproduce their behaviours. Works in this direction (Goldberg & Mataric, 1999) showed the possibility for multi-robot systems to carry out the collective behaviours such as group displacement, gathering, dispersion, foraging. Studies modelling collective behaviour of more advanced animals such as wolves herd showed that the results for archaic animal did not successfully apply in that case. One will also quote the Animatlab approach which conceives simulated or real artificial systems named "animats" whose behaviours exhibit some animal features (Hallam & al., 2002); (Hallam & al., 2004).

2.2 Systems dealing with the communications

The word "communication" can be understood in two different ways: communication between robots or communication between a user and the multi-robot system. Communications between the various entities of a multi-robot system are a crucial point. One often makes the difference between explicit communication, which is a relational operation between an entity and one or more others, and implicit communication ("through the world") in which an entity broadcasts a message which will be received by all others entities.
The problems involved in the user-system communication are tackled in (Jones & Rock, 2002). The aimed application is the use of robots team in the space construction industry. Dialogues are led by the user with a community of agents through a series of implicit and explicit questions on the robots and the environment state. The operator plays a significant role in the scheduling of the robots tasks. Authors underline the difficulties to:

- establish the structure and the range of the dialogue,
- create an infrastructure which allows for the system/robots to conduct a dialogue with the user,
- determine the methods which can take into account the subjacent social aspect in this kind of dialogue,
- develop an interface which allows the user to dialogue with the system.

In (Fong & al., 2001), authors recommend adapting the autonomy and the human-system interaction to the situation and to the user. According to these authors, part of the decision-making process, which is most of time not structured, must remain in the human domain, in particular because the robots remain very limited for the high level perceptive functions, such as objects recognition and "human-like" interpretation of the environment. Their approach tends to treat the robot not like a tool but like a partner.

2.3 Systems directed towards architectures design, tasks allocation and control

These systems deal with problems common to the decentralized systems: tasks allocation, tasks planning, communication system design, homogeneity or heterogeneity of the robots, delegation of authority, global coherence and local actions... In (Iocchi & al., 2001), multi-robot systems are initially shown like a particular case of multi-agents systems. The authors insist on the fact that a multi-robot system has specific constraints because of the immersion of the agents in a real environment.
Other works as those of Rybski (Rybski & al., 2002) present a software architecture intended for the control of a team of miniature robots. Their low on-board calculation capacity calls upon distant calculators. The used algorithm tries to dynamically allocate the resources to the robots according to their needs and to the evolution of the tasks they are carrying out. Tasks allocation is also discussed in the work of Gerkey and Mataric (Mataric & al., 2002) (Gerkey & Mataric, 2003). This article presents a strategy for tasks allocation by using a form of negotiation to optimize the use of the robot's resources. Experimental results are presented for a mission in which an object must be moved.

2.4 Systems for displacements coordination

In the field of displacements coordination of the various robots inside a formation, the main directions of study are the trajectories planning, the generation and the keeping of the formation as well as the traffic control such as they are defined in (Yu & al., 1995). (Das & al., 2002) describe a framework for the co-operative control of a robots group. Authors are interested in the applications of co-operative handling in which a semi-rigid formation is necessary to transport an object. (Tan & Xi, 2004) presents a distributed algorithm for the co-operation and the redeployment of a network of sensors embarked on mobile robots. (Yamashita & al., 2000, 2003) propose a method for movement planning of a robots team for the collective transport of an object in a 3D environment. This task raises various problems like the obstacles avoidance and the stability of the transported object.

2.5 Field of assistance supply for dependent people

If the field of assistance supply to dependent people using mobile robotics, this help is considered by a better ergonomics of the robot and an extended instrumentation of the robot (the robot can be, for example, the handicapped's wheelchair). Without being exhaustive, one can mention the works of the French multidisciplinary national group for the assistance to handicapped people (IFRATH) in which various problems on the man-machine co-operation and the co-operation between robots are studied.

In the ARPH project (Colle & al., 2002), the mobile robot equipped with an arm manipulator is intended to bring an assistance to the handicapped person. This system must help the disabled person to carry out functions of the everyday life: to grip, collect, carry and move. To achieve a task, the person cooperates with the assistance system, each one bringing its own competences and capacities. This collaboration has as main benefit the limitation of the system complexity. The system does not make "instead of" but implies the person at various degrees in the realization of the required service.

VAHM is a prototype of wheelchair mainly intended to help of the handicapped people for to control a conventional wheelchair (Bourhis & al., 2001). In this project, special efforts were made to analyse the user needs and thus to equip the wheelchair with an adapted graphical interface.

Projects proposing the implementation of a team of robots for people assistance are to our knowledge very few. Nevertheless, one will point out works that, without having for goal the assistance of handicapped people, are interested in the aspects bound to:

- the way in which a user can make a request to the system,
- a development environment of low-level behaviours in a multi-robot system,
- a system assigning tasks among a set of autonomous robots,
- several works on the interaction between a user and a multi-robot system.

We propose in the following section a design framework for a participative multi-robot system.

3. Participative multi-robots assistance system

The main objective of this work is to model a system allowing a user and more particularly a disabled person, to give a mission to a team of robots and to determine the whole process leading to its execution.

This work, in order to be realizable, has some limits and constraints:

- Experiments take place on a group of 5 compact, low cost, heterogeneous robots embarking only the minimum processing power and a fixed part managing the heavy computation and the information storage.
- The environment in which will move and operate the mobile robots is an indoor structured environment. Moreover the system has a model of the environment, i.e. a map of the places including all the obstacles and objects the robot will interact with.
- The environment is endowed with a house automation installation allowing the control of the domestic appliances (example: opening the door of the refrigerator).
- The user is part of the system; he/she may, to various degrees, act on it, accept or reject its decisions.

The user has at his/her disposal:

- the knowledge of the apartment, robots and an external view of their possibilities,
- a number of missions of displacement to be carried out like: go to, go towards, return, stop, take, put, gather, and of domestic missions like: bring closer, move away, move an object by pushing it or by pulling it, move an object by collecting it.
- a report on the state of the system.

We approach in this section various concepts which seem important for the study of the human multi-robot systems interaction. We discuss these concepts in order to define our approach and propose our participative, incremental, interactive model with an active help to the specification of a mission to a group of robots.

3.1 Usability

The concept of usability expresses the facility with which a user learns how to operate, to interpret the results of a system. It is carried out that a better adaptation of the machines to the users increases productivity, performances and reactivity.

A significant aspect of the concept of usability is the affordance. Affordance provides the user with a simple access to the actions allowed with a particular object. Controlling a mobile robot is done, for example, through an interface where one selects the zone to be reached by the robot. This possibility is, in general, coded in the interface at the time of its design. However obstacles or a breakdown of one of the robot can prevent it from reaching the indicated zone. An approach using the affordance concept would dynamically adapt according to the capacities and the state of the robot: at every moment, the user is informed of the exact list of the robot's allowed actions. Masking the impossible actions simplifies the work of the user by presenting only relevant information to him/her but can also harm the total comprehension of the state of the environment. This absence of information disturbs, according to us, the creation of a mental model of the progress of the action in the user's mind. For these reasons, a more didactical approach where all the possibilities are presented to the user will be adopted.

3.2 System appropriation

In (Rybarzcyk, 2004), the author explains that the nature of the adaptation of the human to the machine is strongly dependent on the degree of operating differences between these two entities (Piaget, 1936). Thus, if the difference is small, we are in the case of a familiar situation. Under this condition, the adaptation is carried out by a process with dominant of

assimilation[1], i.e. by generalization of the initial schemes relevant for the control of the situation or of the system. Conversely, in the event of a very different operation, the process of adaptation[2] becomes the dominating one (figure 1).
The process of accommodation leads to the transformation and the reorganization of the available action schemes which gradually produce new compositions of schemes allowing the renewed and reproducible control of new classes of situations. It comes out from these observations that the question of the difference between schemes and initial representations of the subjects and schemes and representations necessary to control the machine is crucial in the field of the ergonomic design of the system.
Within this framework, two opposite options are possible. The first one is to seek how to reduce the difference between the spontaneous operator's schemes and those appropriate to the control of the machine. This way consists in regarding the machine as a extension of the user's sensorimotor functions. When such a projection is relevant, the user tends to allot its own characteristics and properties to the system. According to this approach, we have to build machines whose appropriation will be made through a process with dominant of assimilation.

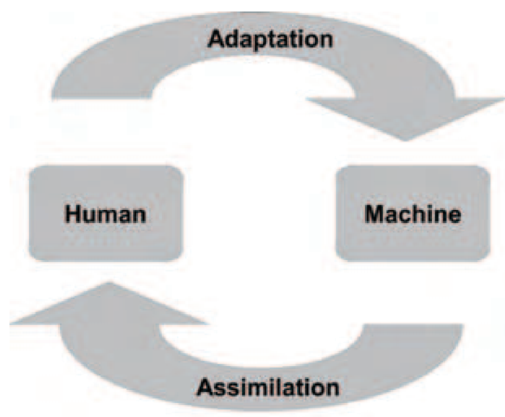


Figure 1. Application of the Piaget model to the man-system co-operation

In our case it does not appear that an identification of the user with the system is possible so much the difference is significant. Moreover, how to imagine a projection of the user in his/her individuality in the plurality of the controlled robots? This is why a second option which takes into account this difference appears much more adapted. Consequently, the design will seek to highlight this difference in order to ease its conceptualization by the subject through a mental model of the teleoperated operations.

3.3 Mental representation

A team leader carries out mental models of the course of the mission he supervises, of the expected results and of the capacities of each team member. This mental representation is used to efficiently allot the various sub-tasks of the on-going mission. In the case of a team

[1] An "action scheme" is the structured set of the generalizable properties of the action which make it possible to repeat the same action or to apply it to new contents. It is by the process of assimilation that the schemes of action will evolve.

[2] Adaptation corresponds to a modification of the organization to adapt to the external conditions. The process of adaptation is used for enriching or widening an action scheme and making it more flexible.

of robots, the operator will probably consider the robots at his disposal as agents allowing him to interact with the environment. The possibilities of each robot being limited to the actions this robot can do at every moment in its environment, we thus underline the need for maintaining a permanent correspondence between the mental representation and the real environment of the robots. However, the system must take into account low level requirements. These constraints lead to a solution differing from the user's mental representation. In this case, the system needs an interface allowing the user-system interaction for the modification of the expression the request (Jones & Synder, 2002).
We propose that the transformation of the mental representation is made in several stages according to a "top-down" process. Initially, the system will take into account the request, analyse it and, if required, propose modifications. If they are accepted, it will build a simulation of the execution and propose it to the user. This one will compare it with its mental representation. If the simulation is accepted, a second stage will transform the diagram into execution in the real environment. We are interested here only by the first stage. We propose to show in section 0 how we partly solved the problem by means of a participative approach with a multi-robot service system.

3.4 User's available information

What information the user has to know in order to have a good comprehension of what occurs during the mission? Several elements appear significant among which:

- in order to simultaneously operate several robots, a general view of the scene is used: a 2D top view (x,y) of the robots environment is adopted to simplify the handling of the interface by the user.
- a short summary of the characteristics of each robot and their current state must be available permanently.
- the user is "in the mission creation loop". Its construction is interactive thanks to a dialogue with the user in order to inform him of his errors or to suggest solutions. The system must provide a textual and graphical feedback to the user as soon as possible. This illustrates the participative aspect of the model.

3.5 Description level

The higher level of a robot control system contains mainly functions like training, reasoning and decision. Its role is to split up the user's requests into secondary tasks and to supervise their implementation. Abstract descriptions of the tasks provided by the user pass through the man-machine interface and are transmitted to a planner. This latter split up the abstract tasks in a comprehensible form by the intermediate control level. The intermediate level allows execution of the behaviours, the primitive actions and supervises the control of the robot subsystems. Low control levels contain the robot sensors and effectors controllers. On the highest level, descriptions and goals are expressed by the abstract terms of the language such as "brings me the blue cube". At the intermediate level, goals are split up into sub-goals but can still contain abstract terms. In lowest levels, descriptions are without any abstraction. The system must allow variable levels of description to allow the users to be more or less precise in their request according to their ease and to their degree of familiarisation with the system. Moreover, the user can delegate the secondary decisions to the system. Thus the choice of the robots involved in the mission can be considered as secondary. The system must also allow a degree of flexibility in the formulation of the mission and allow omission of some intermediate stages. In our case the request is entered through a mission editor located in the highest level.

3.6 General model

Let us recall here the various points highlighted in the above sections, aspects that will guide the modelling:

- the provided service seen as a collaboration between the user and the group of robots,
- the use of a didactical version of the affordance,
- the research of a variable level description of the service request,
- the quantity and the type of information continuously returned to the user,
- the need to maintain a mental representation of the action in coherence with the mission.

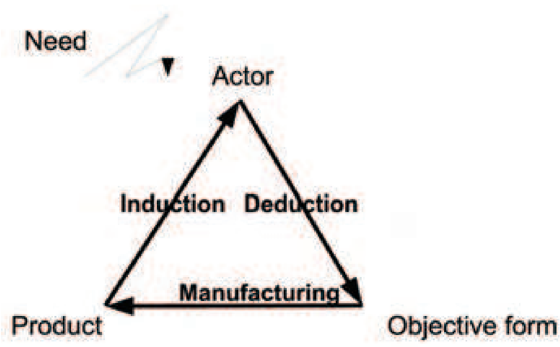


Figure 2. Modelling of the approach

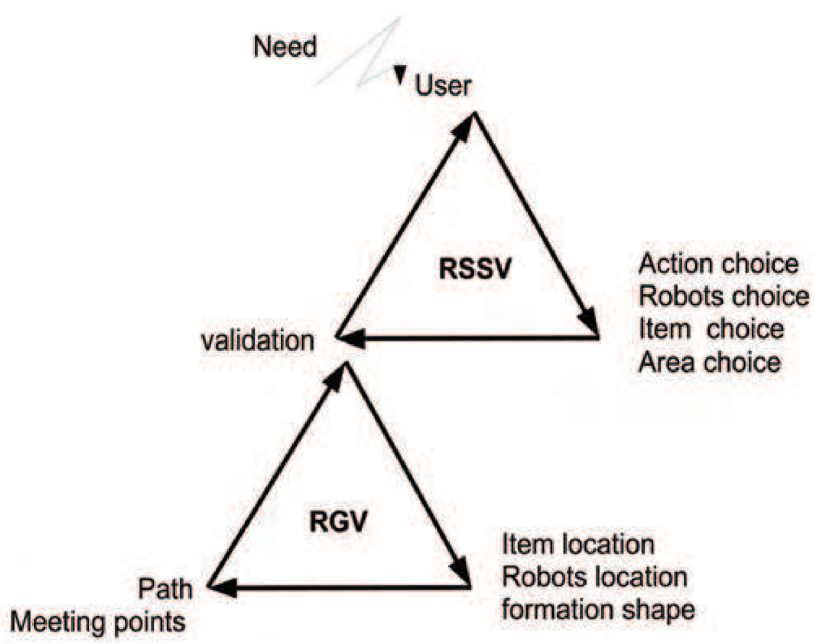


Figure 3. Processing levels

In agreement with these points, we propose in this section a model which calls upon several levels of abstraction. The participative model of this system is given in figure 2. We consider, in this diagram:

- the actor: active element submitted to the stimulus (the user and his request, for example),
- the objective form: object conceived (in the actor's mind, if this one is human, it can be described as a program if the actor is a compiler),
- the product: final state, element satisfying and corresponding to the stimulus (it can be a process whose execution will carry out the stimulus).

The deduction process creates the objective form, consequence drawn from a reasoning, which is calling upon a knowledge. The manufacturing transition transforms the objective form in its material form: the product. The return, which goes from the object to the actor, proceeds from the induction since it goes from the individual level toward a more general entity. We propose to apply recursively this general model to the mission editor.
Our approach is presented in figure 3. We separate two distinct levels:

- RSSV level, for Request Syntactic and Semantic Validation,
- RGV level, for Request Geometrical Validation.

At the RSSV level, the actor is the user; the objective form corresponds to the expression of the request. The deduction is carried out by the user following a service need. The objective form is a description of the robots missions, for example: "the robot *Its_Name* will make *Name_Action*", "the robots will go to *Place*". Manufacturing validates the choice of the robots, supplements and corrects the request by taking into account the constraints. The product, i.e. a new formulation of the mission, is presented to the user.
At the RGV level, the actor element is the RSSV, deduction extracts the relevant parameters for the geometrical checker from the formulation of the mission treated by the RSSV (the product). The manufacturing transition will set up the routes in the environment according to the formations that the team of robots must adopt for the transport of an object. The collective behaviours, such as they are defined in (Goldberg & Mataric, 1999), are directly associated to the various parts of the road to follow. These routes are then proposed to the user.
In the following sections we analyse the functionalities of the RSSV and RGV levels. We then propose solutions we implemented to create the functionalities required by the two levels of the participative model.

4. Request syntactic and semantic verificator

4.1 Object modelling

The first stage consists in modelling the various objects which will play a role in the system. Modelling emphasizes two main categories: the robots class and the objects class (figures 4 and 5).
Modelling of the robots is mainly guided by their features in terms of effectors and sensors. The modelling of the class ITEM makes it possible to describe the objects of the environment. There are two types of objects from which derive all the objects in the environment: the ITEM_ACTIVE and the ITEM_PASSIVE according to whether they are simple obstacles on which no action is possible or objects with which the robots can interact. The class AREA contains the various zones of the environment.
The number of rules evolves according to the complexity of the model of the environment. Currently, the rule base contains about sixty entries. In the following sections, we will describe the main rules classified in several groups. One will detail successively the rules that:

- check the number of arguments,

- solve conflicts between arguments,
- check the existence of ways between two rooms,
- generate the missing stages and check the state of the robots,
- check the compatibility between effector, action and item,
- check the number of robots for the transport of the object and choose the adapted formation,
- indicates the robots ready to achieve the task {*take*}.

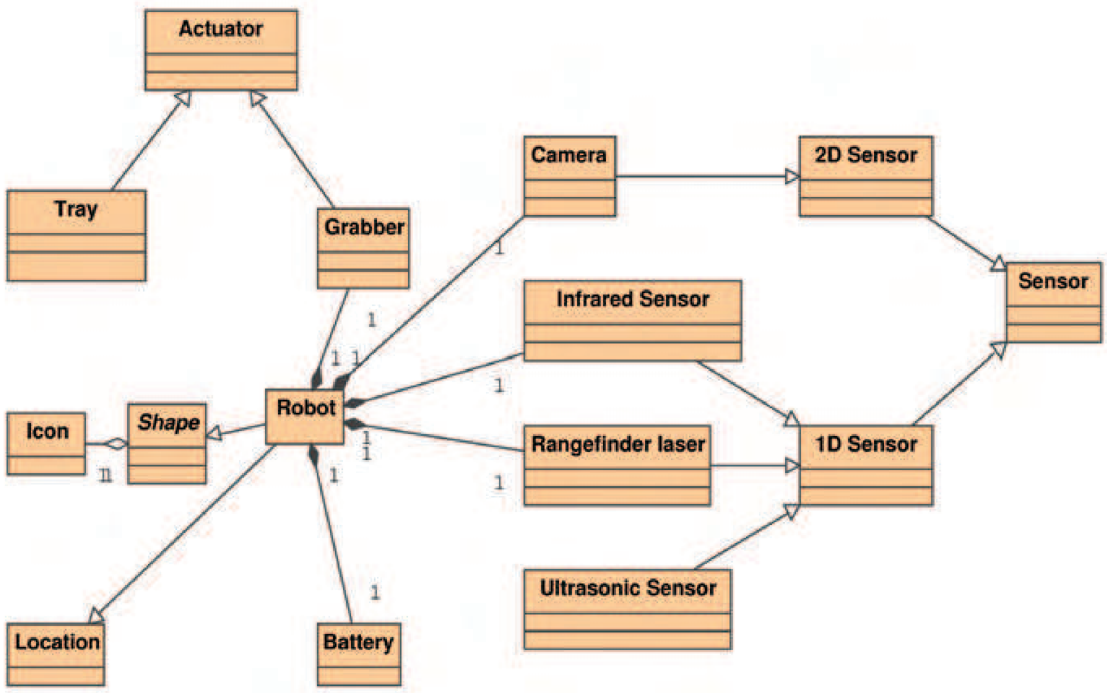
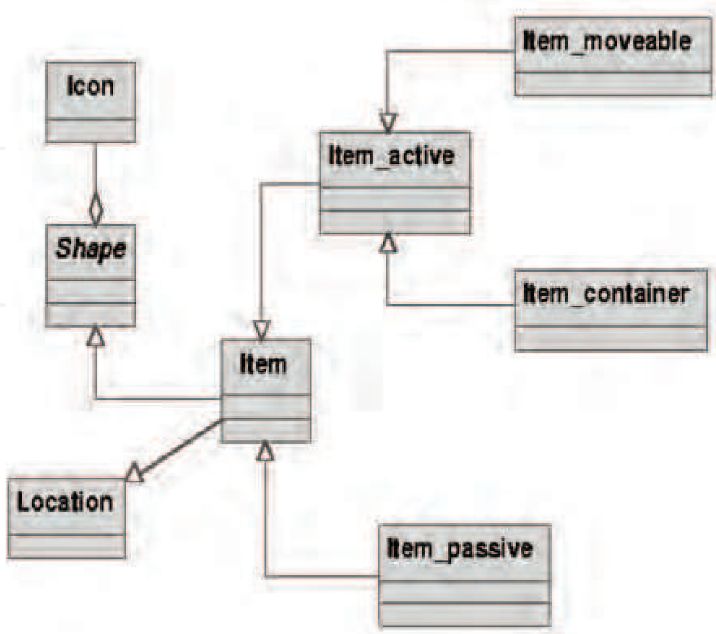


Figure 4. ROBOT class



Refrigerator

is_a_transportable = false
is_a_container = true
possible_action = open
needed_effectors = gripper
needed_sensor = camera
location = kitchen

Figure 5. ITEM class and item example

Before approaching the rules description, we briefly describe the format of the various elements specifying the mission. The user's request is represented by a succession of facts {chosen_X ?} in which X and ? can take various values:

- {chosen_action ?action} where ?action can have a value among {goto, gather, take, could, search}.
- {chosen_robot ?robot} where ?robot can have a value among {america, africa, asia, europe, oceania} (names of the five robots).
- {chosen_object ?object} where ?object can have any value of the type ITEM_ACTIVE.
- {chosen_area ?area} where ?area can have any value of the type AREA.
- the absence of a fact {chosen_X ?} systematically causes the insertion of the fact {missing_X} (X represents one of the parameters: robot, object or area).
- in addition, to each {chosen_X ?} describing a user's choice, corresponds {computed_X ?}, value of the parameter X calculated by the inference engine which will contain the arguments corrected or deduced by the system.

4.2 Request syntactic checking: Arguments number checking and arguments conflicts

When the various parameters are entered by the user, the inference engine starts by checking the number of received arguments. Each time a fact of type {chosen_X} is missing, the system inserts the fact {missing_X}. The following paragraphs describe the reaction of the system. Some examples of actions are given.

Case of action *goto*

- {missing_robot}: the system can choose by itself the robot which will carry out displacement. The choice of the robot will be done according to the mode in progress (parallel or sequential, section 0),
- {missing_area}: the presence of this fact blocks execution,
- {missing_object}: the presence of this fact is ignored,
- {chosen_object ?object}: the position of the object is extracted, starting from the data base, and inserted as a {computed_area}. In the event of conflict with a {chosen_area}, the system requires the user to solve the ambiguity on the choice of the robots destination.

Case of action *take*

- {missing_robot}: the system chooses one or more robots compatible with the action take on the object,
- {missing_area}: the presence of this fact is ignored,
- {missing_object}: the position of the object is extracted, starting from the data base, and inserted as a {computed_area}. In the event of conflict with a {chosen_area}, the system requires the user to solve ambiguity on the choice of the destination of the robots.

Case of action *put*

- {missing_robot}: if {chosen_object} exists, the robots which transport the object are used to create {computed_robot}. If not, the system requires the user to specify the request,
- {missing_area}: the current position of the robot is used to add {computed_area},
- {missing_object}: if {chosen_robot} exists, the object transported by the robot(s) is used to create {computed_object}. If not, the user is asked to be more precise in its request.

Table 1 recapitulates the links between the selected action and the type of required arguments. Any compulsory missing argument produces an error which results in a message inviting the user to be more precise and to provide the missing argument. Certain actions as action *put* (table 2) are a little bit particular and use a group of specific rules: the user can either specify the robot that transports or the transported object.

Action	Goto	Gather	Take	Search
Optional arguments	robots_list	robots_list area	robot	robots_list area
Compulsory arguments	area		Transportable object	Object

Table 1. Actions/arguments table

Put	Optional arguments	Compulsory arguments
case 1	robot	carried object
case 2	carried object	Robot

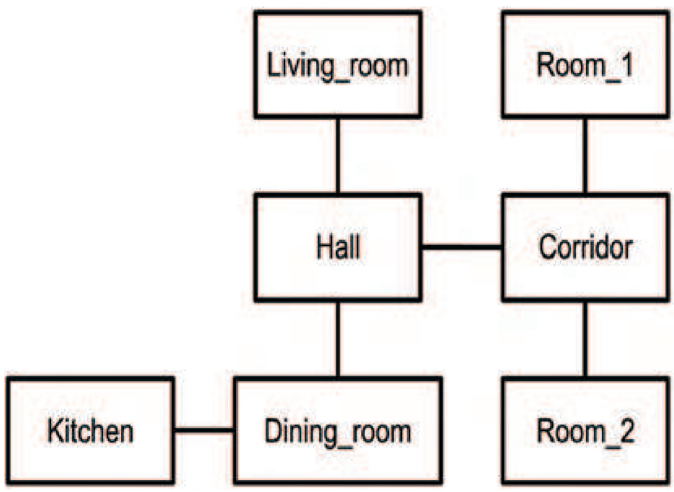
Table 2. *Particular case: action put*

Arguments conflicts can occur in particular for action *put* where the robot and object-choice do not correspond (the robot does not transport the object indicated by the user). Some of these conflicts are solved in an automatic way: the conflict of argument is solved by the creation of an action *take* for the object indicated before carrying out the action *put*.

4.3 Environment topology checking

One of the roles of the RSSV is to check the accessibility of the activity area by the selected robots. The initial base of facts contains the list of the paths between two close parts in the form {path area1 area2}, {path area2 area3} (Paths such as {path area1 area3} are automatically deduced by chaining rules). It is the only representation of the environment available to CLIPS[3] (figure 6). CLIPS thus checks the existence of a path between the current position of each robot and the position to be reached. If the current position of the robot does allow reaching the final position, the system chooses another robot.

[3] CLIPS is an expert system development environment. See http://www.ghg.net/clips/CLIPS.html for details



{path living_room hall}
{path hall dining_room}
{path kitchen dining_room}
{path room1 corridor}
{path hall corridor}
{path corridor room2}

Figure 6. Topology of the environment known by the RSSV

4.4 Checking robots status and action/object/robot compatibility

The validation of the request checks the availability of the robots chosen by the user. Two variables manage the status of the robots, one indicates if the robot operates correctly, the other indicates if it is in use. The system will preferably choose robots which are not in use in order to be able to start the task immediately. Section 0 discusses some details regarding the possibilities of automatic robot allocation offered by the RSSV.

The modelling of the sensors and the effector includes the list of the achievable actions. Thus a position encoder is necessary to the action *goto*, an effector of the arm type is necessary to the action *take*. Moreover, the description of each object contains the list of its possible actions. The RSSV checks if the selected robots have the features necessary to achieve the required action. In the opposite case, a warning is sent to the user and the RSSV switches to the robot automatic allocation mode (section 0). In addition, for actions not possible with certain types of objects (for example, the action *take* must be mandatory carried out on object of TRANSPORTABLE type.) the system signals the inconsistency of the request to the user so that he may correct it.

4.5 Robots number checking and formation choice

The services offered by the group of robot are among others, the transport of objects and, in particular, collective transport of bulky objects. The number of robots necessary to act on an object is fixed during the modelling phase (section 0) in the description of the object. Thus there must be as many facts {chosen_robot} than the number of robots necessary to achieve the task. Specific rules manage the selection of the groups of robots. The following example illustrates these two RSSV functionalities: initially, the user makes the request {*take* chair America} the system then adds an additional robot (Asia for example) for this task because the action *take* chair requires two robots. In a second step, the user makes the request {*goto* America living_room}. To satisfy this request the system will create two actions *goto* with the robots America and Asia. The modelling of the object contains also the formation to be adopted during its transport. The geometry of the formation is not used by the RSSV and will be transmitted to the RGV to check the possibilities of passage of the formation.

4.6 Robots automatic choice

When the user does not specify a robot for a given task, or when a robot is missing in order to achieve a task, or when the selected robot cannot achieve the task (robot out of service, ...), the RSSV can select robots in an automatic way. To each robot is associated a coefficient called *usage_score*. This concept was introduced in order to be able to classify the robots according to a criterion of execution cost. The calculation of the *usage_score* takes into account the physical features of the robots in terms of embarked sensors and effectors. The value of each robot's feature was adjusted following several tests. The *usage_score* takes also into account the current state of the robot and will be higher if the robot is in use. When the user does not specify the robots that must take part in the action, the system chooses the robot that has the necessary functionalities and the lowest *usage_score*. Two operating modes can be used by the inference mechanism: the parallel and the sequential modes.

In the parallel mode, the system endeavours to parallelize the tasks carried out by the robots by choosing a new robot for each new task (with respect to the lowest *usage_score*) criterion. In the sequential mode, the system will help the user to build the mission step by step. In opposition to the parallel mode, the system tries to assign the maximum of tasks to the same robot in order to make it possible to the user to build a complex mission. In the current state of the project, the system cannot change the affected robot for the preceding tasks. The case occurs when the user inserts at stage N an action which cannot be achieved by the robot selected automatically for the N-1 first tasks. An evolution of the system will allow each task to have an influence on the choice of the robot of the preceding tasks. Lastly, it is planned that the system can switch from one mode to another during the edition of a mission, in order to benefit from both edition modes according to the context.

4.7 Example of operation

This section illustrates a realistic scenario in which the user wants a soda can from refrigerator. The request of the user thus takes the simple form: {*take* can}.

The figure 7 illustrates the operation completed by the RSSV. The following verifications are made:

- "can" item is known by the system,
- "can" item is compatible with the action *take*,
- "can" item is located the container "refrigerator",
- type and numbers of robots necessary for action *take* on item "can",
- availability of the robots,
- type and numbers of robots necessary for action *take* on item "can" from item "refrigerator",
- the container "refrigerator" is reachable.

Once all these constraints checked, a robot (America), compatible and free, is chosen, an intermediate action {*goto* America kitchen} is created and the result is displayed to the user. The following section presents the second level of the system.

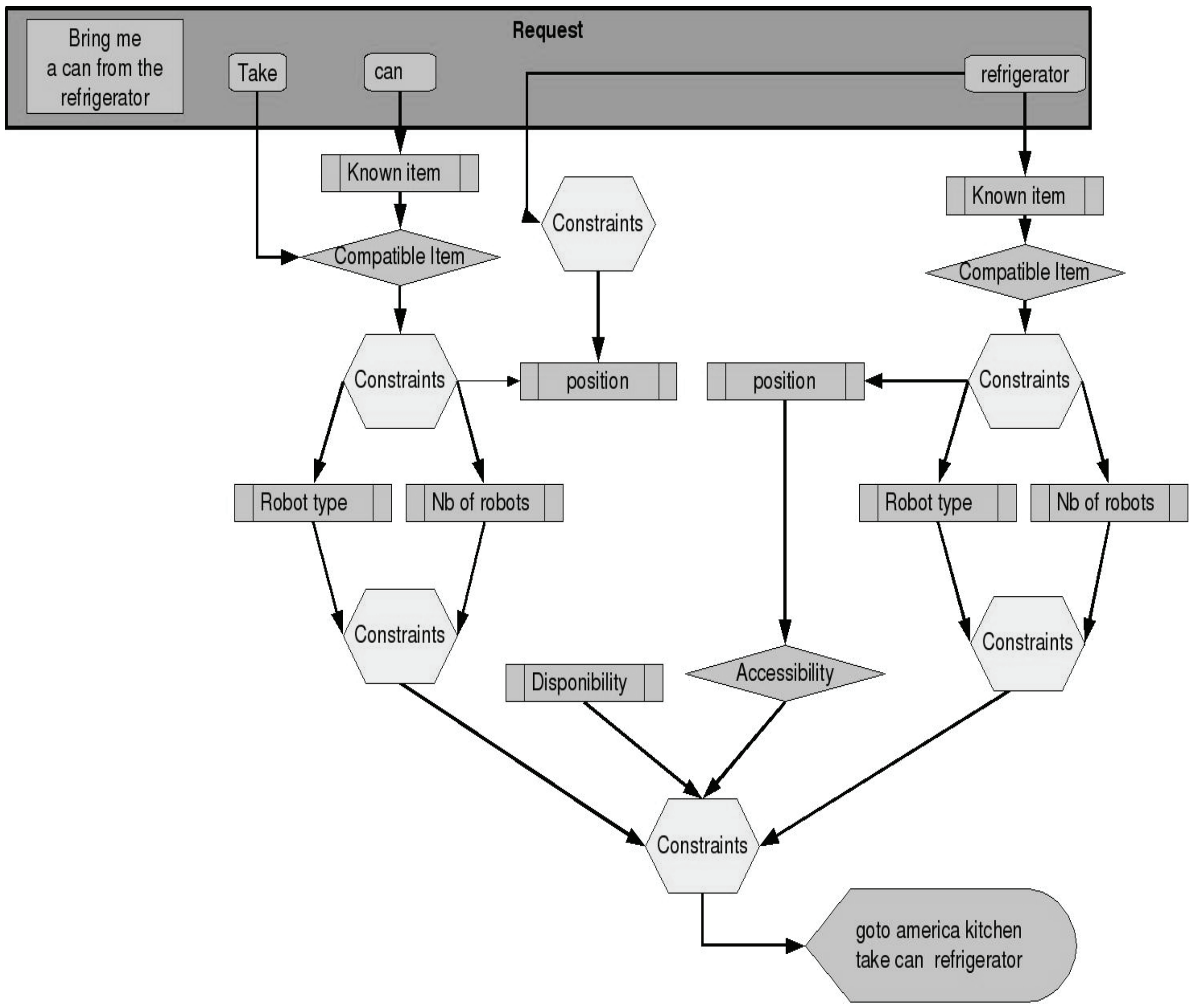


Figure 7. *Example of request*

5. Request Geometrical Validation

This section presents the Request Geometrical Validation (RGV): the constraints, the simplifying assumptions, the stages allowing the validation of the robots formation between the selected points.

First let's recall the functionalities the RGV module must provide:

- give an outline of the robots trajectories to the user,
- find the meeting points for group navigation,
- validate or reject the passages for the formations according to the meeting points.

In the discrete representation of the environment that was chosen, the environment is divided into cells of fixed size. A cell can be either in an obstacle or in a open space. The cells table representing the environment is dynamically created on each program start from information stored in a data base. A map representing the accessible points of the environment is illustrated in figure 8.

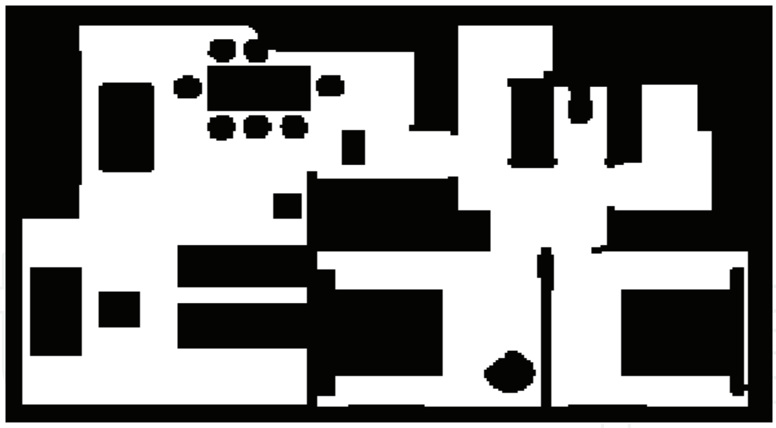


Figure 8. Environment occupation map

Figure 9 shows the various steps which lead to the geometrical checking of the robots formation path. Each step will be detailed in the following paragraphs.

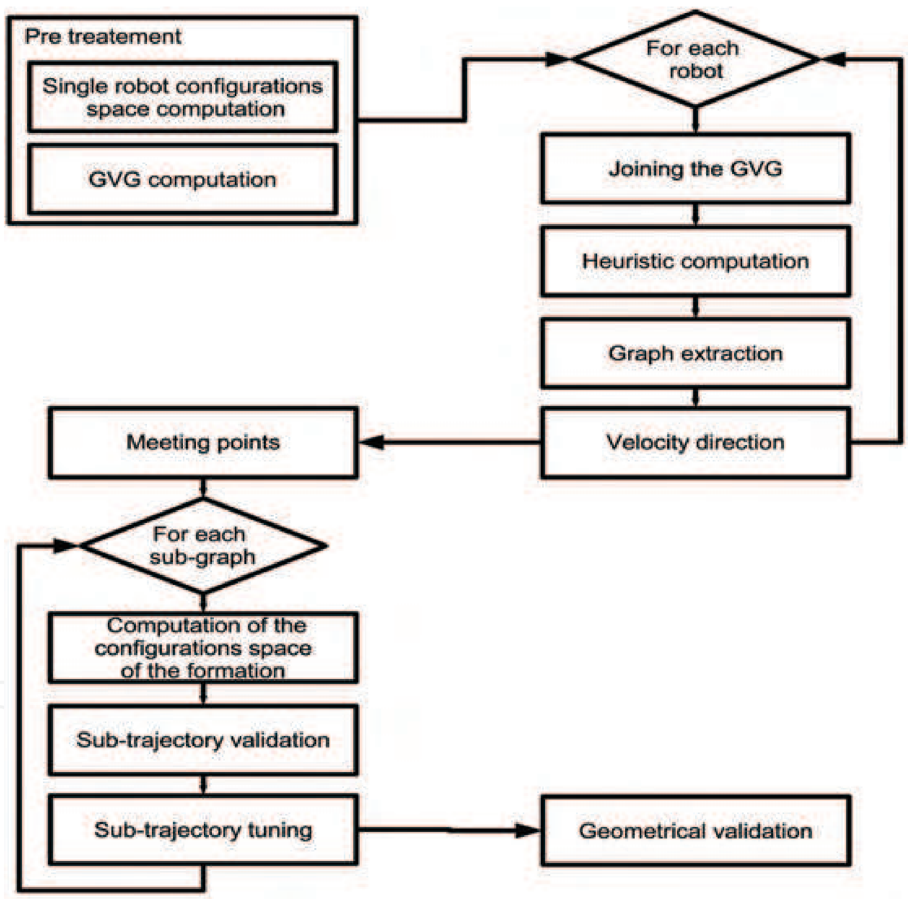


Figure 9. Processing steps leading to the routes geometrical checking

In the following sections, we will denote by:

- C the configurations space,
- C_{open} the set of valid configurations in C,
- $C_{obstacles}$ the set of invalid configurations in C,
- q_i a configuration in C,
- R(q) the set of points occupied by the robot R in configuration q.

5.1 Environment preprocessing: Calculation of the robot configurations space and the GVG

A first off-line environment processing is carried out. It includes two stages: calculation of the configurations space of a robot and calculation of the generalized Voronoï graph in this space.

The robots are cylindrical, they have 3 parameters x,y and θ (x, y: position, θ: orientation). Thus *R(q)* does not depend on θ (the points occupied by the robot do not depend on the parameter θ). The configurations space has only two dimensions isomorphous to the Euclidean plan. The configurations space is then approximated by a dilation algorithm using the d_8 distance. This distance approximation introduces a safety margin around the obstacles.

To connect the initial and final configurations in C_{open}, we use a generalized Voronoï graph (GVG) calculated on C_{open}. The GVG represents the set of the points of C_{open} equidistant of $C_{obstacles}$.

5.2 Individual trajectories processing

Each robot is individually considered to obtain a route in C_{open} between the final and starting configurations.

As the starting and final positions of the robots seldomly match a point on the GVG, Bezier splines join these positions to the nearest GVG point.

From the graph created in section 0, a directed sub-graph corresponding to the valid configurations between the starting and final positions is extracted.

To speed up the graph exploration, the distance between each node of the graph and the target point is evaluated. This evaluation allows the exploration function to choose the next node to be visited while approaching the target. A wavefront algorithm stamps each point of the graph with the distance separating it from the target. This distance takes account of the obstacles and avoids all local minima (figure 10).

Figure 10. Example of the wave front propagation in the environment

From the starting configuration q_d, the GVG nodes are visited following the potentials given by the wave front algorithm. Finally, the valid configurations set between q_d and q_f is obtained for each robot. This valid configurations set q_i is stored in a vector noted $T_{i \in R,j}$.

Each configuration contains only 2 parameters corresponding to x and y. The robot orientation, in each configuration q_i, is added to vector T. All the orientations θ being allowed, the direction of the local tangent is calculated for each position on the trajectory which provides the missing parameter (figure 11).

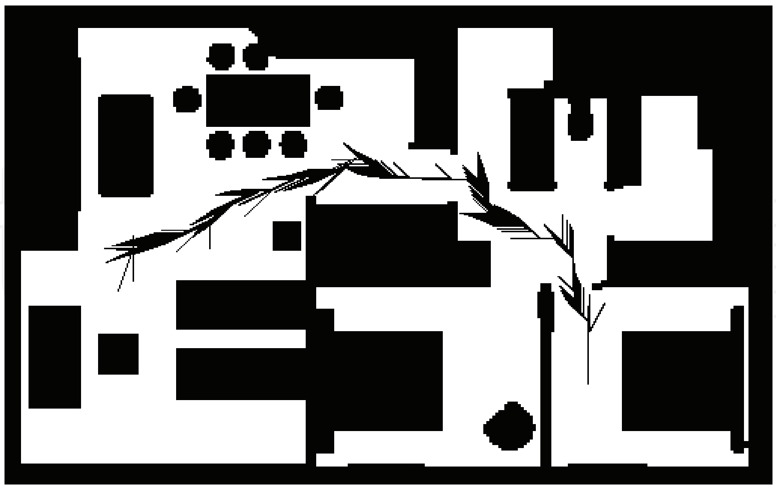


Figure 11. Representation of the speed vector

These stages lead to the construction of a vector set $t_{i \in R,j}$ representing the series of the free configurations between the starting and final points of each robot involved in the mission. The following stages lead to the creation of robots groups for navigation in formation.

5.3 Superposition of individual trajectories}

By superposition of the various individual routes (figures 12 to 14), the meeting points as well as the trajectories segments along which the robots will navigate in formation are extracted.

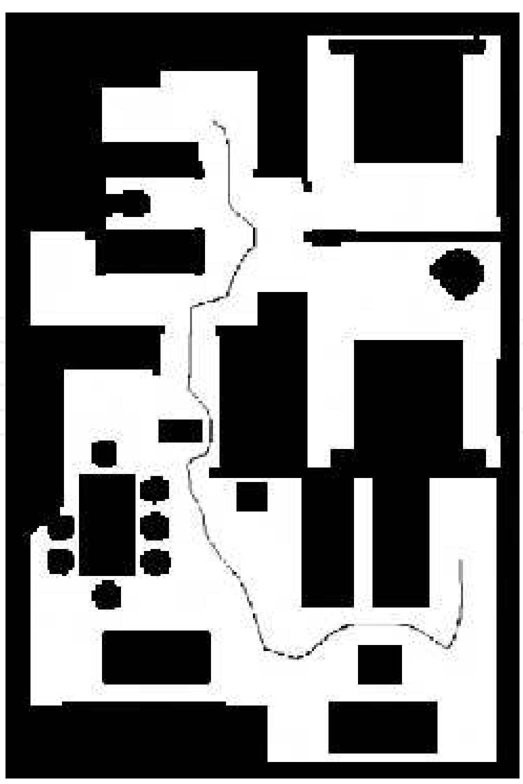


Figure 12. Robot 1 trajectory

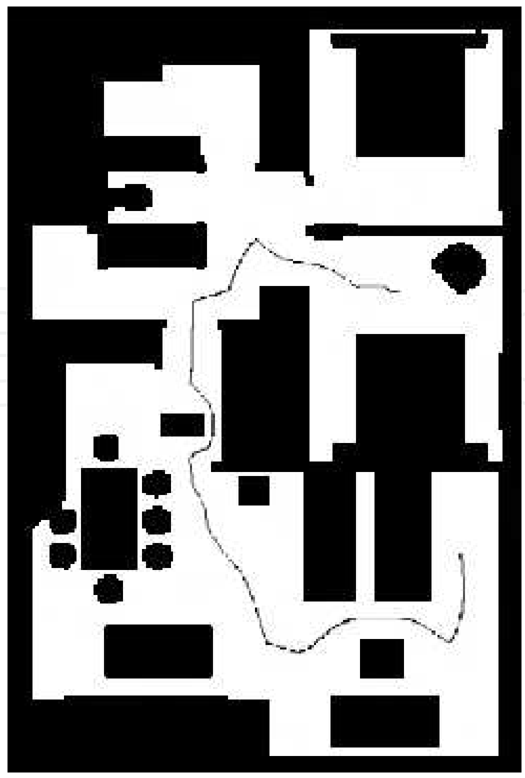


Figure 13. Robot 2 trajectory

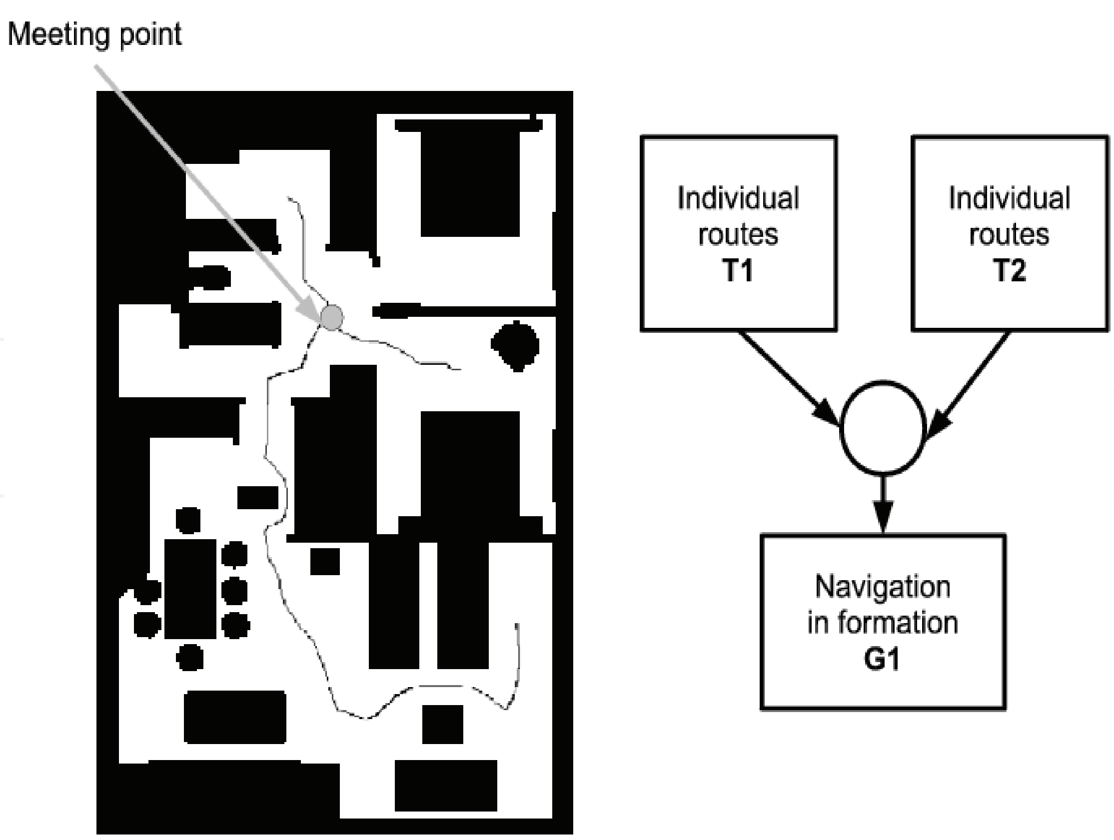


Figure 14. Trajectory of the two robots and navigation graph

Free configurations being taken on the GVG of the configurations space, the vectors of two robots having the same destination contain at least a common configuration. As soon as a common configuration is detected, a convoy formed by the two robots is created starting from this point. The process continues and other robots are added into the convoy. The vector containing the configurations of the formation resulting from the meeting point i is called G_i.

5.4 Adaptation of the sub-trajectories to the formations

The preceding stage created a set of G_i vectors containing the configurations of the robots between two meeting points. The configuration q ($q \in G_i$), initially defined as the configuration of a single robot, is now considered as the configuration of the robots formation whose reference point is its barycentre.
Because of this new status, the G_i vectors can contain invalid formation configurations as they were calculated in C_{open} of a single robot and not in C_{open} of the whole formation. The G_i vector must be modified in order to meet the formation constraint (each configuration must be in C_{open} of the formation) while maintaining the configurations connexity.

5.5 Computing the formation configurations space

Assume R_i is the set of the robots belonging to the formation, $R(q)$ is the set of points occupied by R_i while being in the configuration q and O the set of the points of the transported object. H is defined as $F = O \bigcup_i R_i$. One notes $\hat{H}$ the convex hull containing H.

C_{open} of $\hat{H}$ is included in the configuration space of the formation carrying the object, thus any configuration in $C_{open}/\hat{H}$ is also in C_{open}/H .

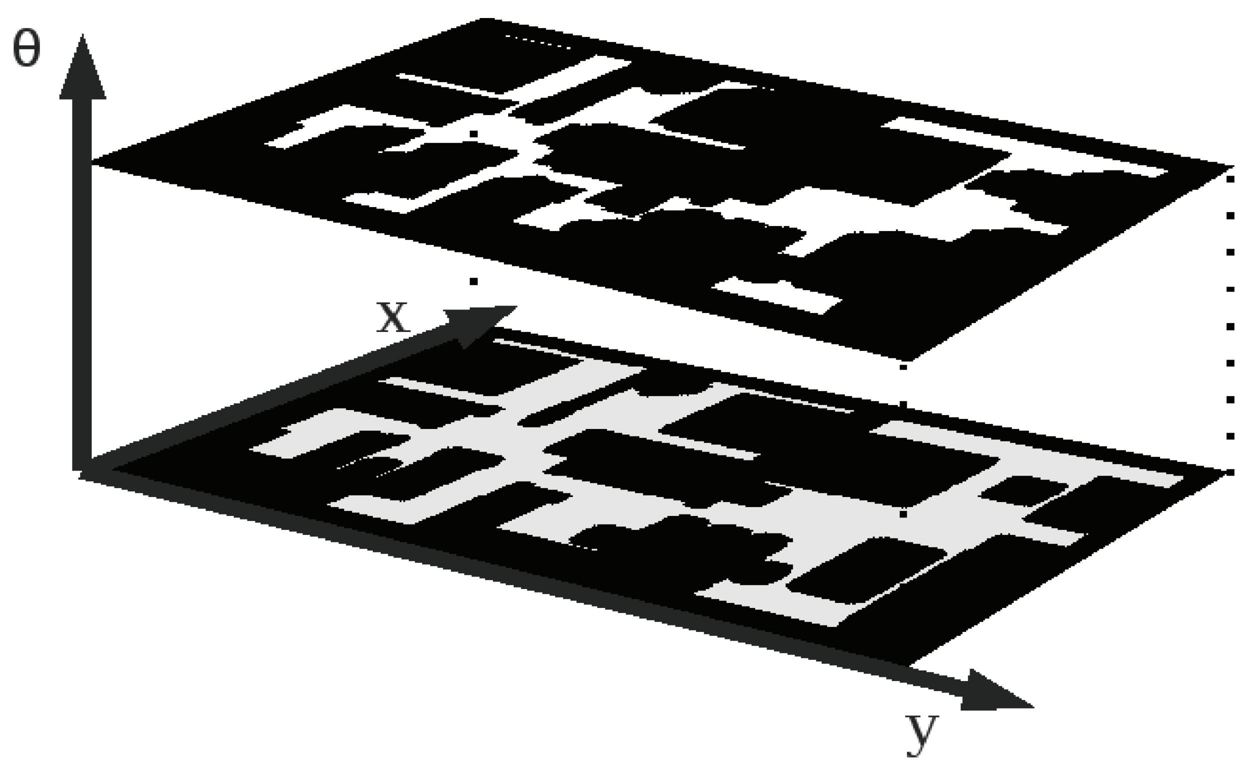


Figure 15. Configuration space of a robots formation. The configurations space is shown for two orientations of the robot formation

Two configurations q_1 and q_2 connected in C_{open}/R could not be connected by a free path in $C_{open}/\hat{H}$. To check the existence of a path between two consecutive meeting points in $C_{open}/\hat{H}$ a propagation algorithm (described in section 0), adapted to a 3 dimensional configurations space, is used. The distances are propagated from the final configuration q_2 until reaching the initial configuration q_1. If the propagation succeeds, q_1 and q_1 are connected in $C_{open}/\hat{H}$ (figure 15). If the propagation fails, the robots formation will not be able to move from q_1 to q_2.

Once the existence of a free path between the two ends of the group trajectory is established in the configurations space of the formation, it is still necessary to check the absence of collisions between the trajectory curve and the $C_{obstacles}$. If a collision occurs, on each collision zone, the trajectory is locally tuned in order to satisfy to the constraints. The tuning consists in a deformation of the original trajectory by selecting a clear path between the point preceding the clash with the obstacle and the point following that same clash. This clear path is also computed thanks to a local front wave algorithm (figure 16). This method ensures the least deformation between the original trajectory and the new one. This deformation according to the three dimensions of the C_{space} corresponds to translations and rotations of the formation in the Euclidean space.

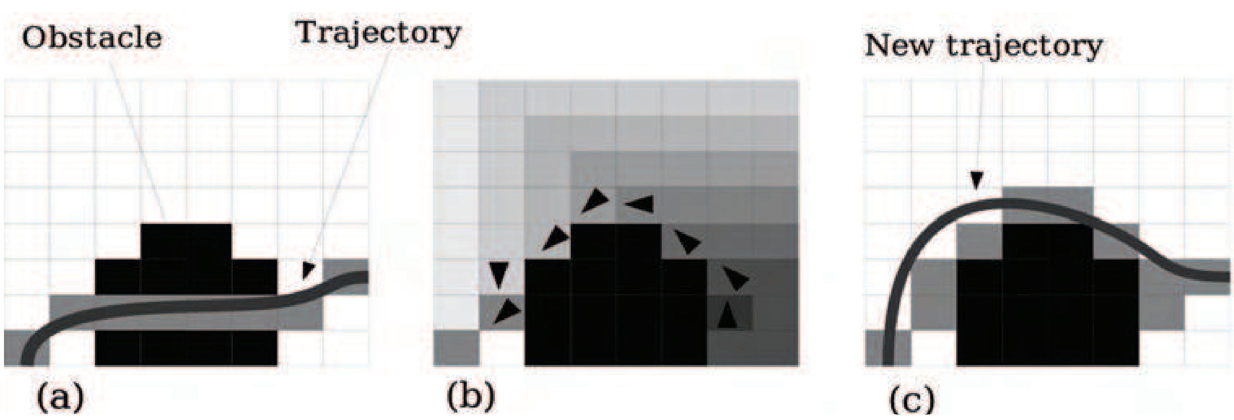


Figure 16. Avoidance of a $C_{obstacle}$

The operation is reiterated on each couple $(q_e, q_s)_i$ belonging to the vector G_i to create the vector $\hat{g}_i$ valid on C_{open}/F. Then, for each vector G_i, the vector $\hat{g}_i$ is calculated.

5.6 Final group trajectory and user interface

Lastly, a series of vectors $\hat{G}_i$ corresponding to the various configurations of the formations adopted by robots after every meeting point is displayed on the user interface (figure 17). This interface is divided into several zones:

- the robots selection zone: each button displays the name and the state of the robot,
- the objects selection zone: the objects retrieved from a database are accessible via a drop-down menu.
- the map zone: represents the environment in which the robots evolve.
- the actions zone: a simple set of buttons attached to the possible actions.
- the information zone: general purpose message zone.
- the RGV zone: RGV control panel.
- the debug zone: access to useful functionalities during program development.

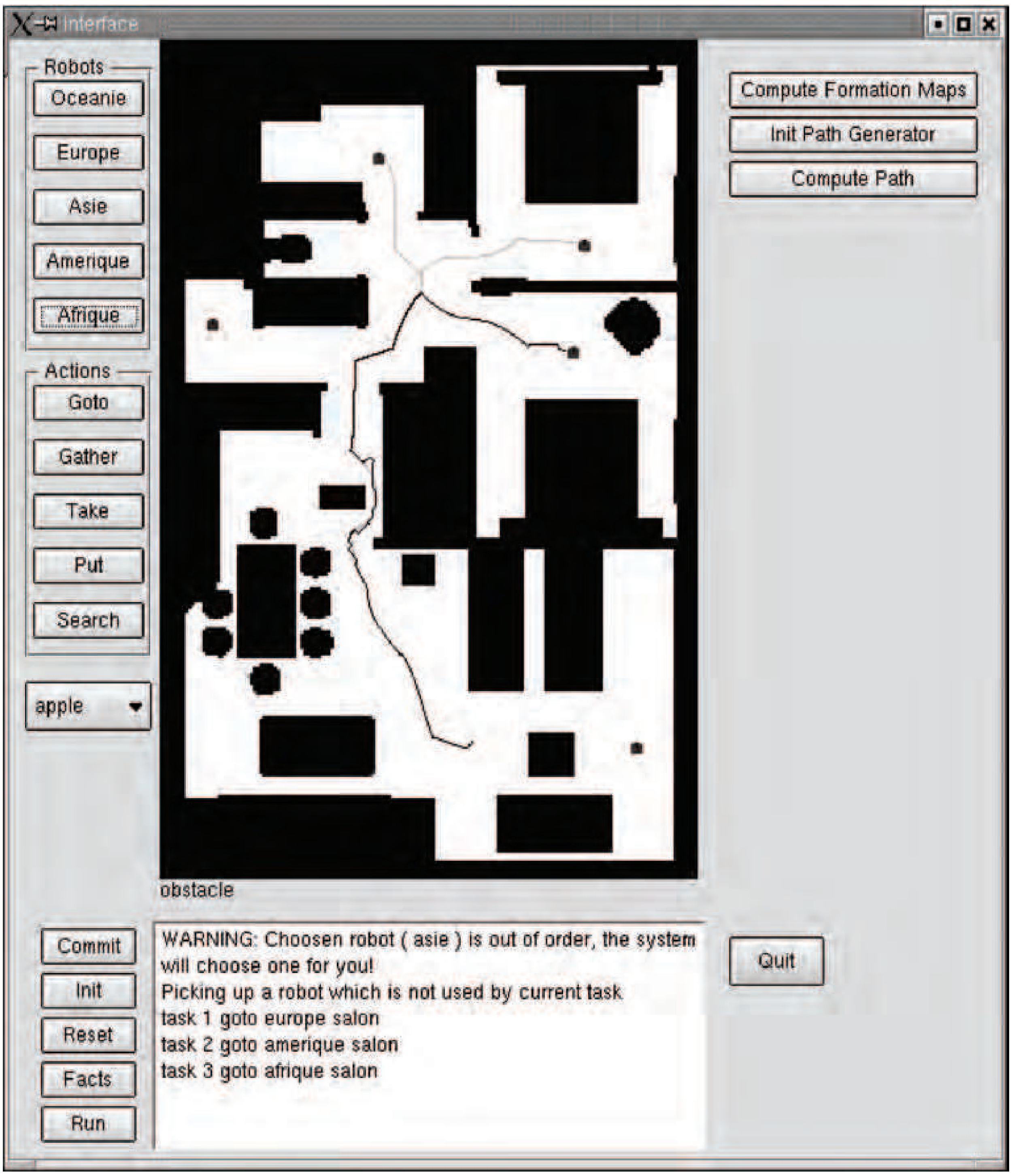


Figure 17. User's interface screenshot

The passages of the formations as well as the transported object must then be validated by the user before execution.

6. Conclusion and prospects

This work aims at modelling the interaction between a user and a multi-robot system in order to give a mission to a team of robots and to determine the set of the driving process leading to its execution.
According to some hypotheses among which:

- the indoor structured environment is known,
- the small number of robots,
- the cost which influences the system architecture,
- the user may act on the system decisions,

the user can request some missions the system is able to accomplish. We mainly insist on the system participative aspect in order to be accepted by the user in spite of its limitations. This participative aspect implies:

- a human-system interface taking care of the communication in a friendly, click and drag, icon based, high-level language,
- the expression of the request proposed as a set of remote services carried out by a reduced number of robots,
- the ability of act during the construction of the solution to the request,
- a way to act directly on the robots and on the mission scene in the event of modification, breakdown or execution failure,
- an information on the system state always available.

The originality of this work stands in our approach of the interaction between the user and the multi-robot system. The required services are seen as a man/machine co-operation where the user delegates some decisions to the system. We qualify our model of participative, incremental, interactive with an active help to the specification of the mission. Participative because the user is involved in the mission creation process. Incremental, because the request is analysed through two sequential levels. Interactive, because a constant dialogue between the user and the system is maintained throughout the construction of the request. We propose solutions for the realization of the functionalities of the two levels of our model. The realization of the first level uses of an inference engine and a rule base. The second level, the geometrical verificator uses trajectory planning methods. The two levels are assembled in a network architecture which includes a database and a graphical user interface to complete the multi-robot mission editor. Experiments allow us to validate the usability of this system. Tests on real robots are ongoing.
Several evolutions prospects and improvements of the system are possible. In one hand, the number of objects and actions must be extended to offer more services to the user. In addition, it is necessary to modify the human-system interface to take into account more complex missions and to allow interactivity during the mission execution. An improvement of the general ergonomics of the interface must also be considered after an evaluation of the system will be made by a handicapped person. Finally the lower levels of the robots control must be set up. Once these levels set up a second phase of study of the editor will be able to take place. In this phase the readjustments of the contents of the database according to the information collected by the robots will be done.

7. References

Ani:02: Hallam, B. & Floreano, D., Hallam, J. & Hayes, G. & Meyer, J.A. (2002). From Animals to Animats 7, *Proceedings of the Seventh International Conference on Simulation of Adaptive Behavior*.

Ani:04: Hallam, J. & Meyer, J.A. & Schaal, S. & Ijspeert, A.J. & Billard, A. & Vijayakumar, S. (eds) (2004). From Animals to Animats 8, *Proceedings of the Eighth International Conference on Simulation of Adaptive Behavior*.

Arai, T. & Ogata, H. & Susuki, T. (1989). Collision avoidance among multiple robots using virtual inpedance. *In Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. pp. 479–485.

Arai, T. & Pagello, E. & Parker, L. (2002). Editorial: Advances in multi-robot systems. *IEEE Trans. Robot. Autom*. 18(5), pp. 655–6617.

Asama, H. & Matsumoto, A. & Ishida, Y. (1989). Design of an autonomous and distributed robot system. *In Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 283–290.

Balch, T. & Parker, L.: (2002). Robot teams: From polymorphism to diversity. Balch, T. & Parker, L. (eds.)

Beni, G. & Hackwood, S. (1992). Stationary waves in cyclic swarms. *In IEEE Int. Symp. on Intelligent Control*, pp. 234–242.

Bonabeau, E. & Theraulaz, G. (2000). *L'intelligence en essaim.*

Bourhis, G. & Horn, O. & Habert, O. & Pruski, A. (2001). An autonomous vehicle for people with motor disabilities. *IEEE Robot. Autom. Mag*. 7(1), pp. 20–28.

Brooks, R. (1986). A robust layered control system for a mobile robot. *J. Robot. Autom*. 2(1), pp. 14–237.

Colle, E. & Rybarczyk, Y. & Hoppenot, P. (2002). ARPH: An assistant robot for disabled people. *In IEEE International Conference on Systems, Man and Cybernetics.*

Das, A. & Fierro, R. & Kumar, V. & Ostrowski, J. & Spletzer, J. & Taylor, C. (2002). A vision-based formation control framework. *IEEE Trans. Robot. Autom*. 18(5), pp. 813–825.

Drogoul, A. & Ferber, J. (1993). From tom-thumb to the dockers: Some experiments with foraging robots. *In From Animals to Animats 2, Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pp. 451–459.

Fong, T. & Thorpe, C. & Baur, C. (2001). Collaboration, dialogue, and human–robot interaction. *In Proceedings of the International Symposium on Robotics Research*, Lorne, Victoria, Australia.

Fukuda, T. & Nakagawa, S. (1987). A dynamically reconfigurable robotic system (concept of a system and optimal configurations. *In Proceedings of IECON'87*, pp. 588–595.

Gerkey, B. & Mataric, M. (2003). A framework for studying multi-robot task allocation. I*n Proceedings of the Second International Naval Research Laboratory Workshop on Multi-Robot Systems*. Washington, District of Columbia, pp. 3367–3373.

Goldberg, D. & Mataric, M. (1999). Coordinating mobile robot group behavior using a model of interaction dynamics. *In Autonomous Agents*. Seattle, Washington, pp. 100–107.

Iocchi, L. & Nardi, D. & Salerno, M. (2001). Reactivity and deliberation: A survey on multi-robot systems.

Jones, H. & Rock, S. (2002). Dialogue-based human–robot interaction for space construction teams. *In Proceedings of the 2002 IEEE Aerospace Conference*, Big Sky, Montana.

Jones, H. & Synder, M. (2002). Supervisory control of multiple robots based on a real-time strategy game interaction paradigm. I*n ACM Computer Supported Cooperative Work.*

Lemay, M. (2003) Systèmes flous pour le contrôle d'une formation de robots.

Mataric, M. & Sukhatme, G. &Ostergaard, E. (2002). Sold!: Auction methods for multi-robot coordination. *IEEE Trans. Robot. Autom. special issue on Advances in Multi-Robot Systems*. 5(18), pp. 758–786 .

Parker, L. (1998*)*. ALLIANCE: An architecture for fault-tolerant multi-robot cooperation. *IEEE Trans. Robot. Autom*. 14(2), pp. 220–240.

Parker, L. (2000). Current state of the art in distributed autonomous mobile robotics. *Distributed Autonomous Robotics Systems.* 18(4), pp. 3–12.

Parker, L. & Bekey, G. & Barhen, J. (2002). Distributed Autonomous Roboti systems, vol. 4. Springer, Berlin Heidelberg, New York.

Piaget, J. (1936). La Naissance de l'Intelligence chez l'Enfant.

Premvuti, S. & Yuta, S. (1989). Considerations on the cooperation of multiple autonomous mobile robots. *In Proceedings of the IEEE Int. Workshop on Intelligent Robots and Systems*, pp. 59–63.

Rybarczyk, Y. (2004). Etude de l'appropriation d'un système de tél´eopération dans l'optique d'une coopération homme-machine. *PhD thesis*, Université d'Evry-Val d'Essonne, Evry, France.

Rybski, P. & Stoeter, S. & Gini, M. & Hougen, D. & Papanikolopoulos, N. (2002). Performance of a distributed robotic system using shared communication channels. *IEEE Trans. Robot. Autom*. 18(5), pp. 211–225.

Schultz, A. & Parker, L. (2002). Multi-robot Systems: From Swarms to Intelligent Automata. Kluwer.

Sellem, P. & Dalgalarrondo, A. (1999). Extension d'une architecture de contrôle derobot mobile à unsystème distribué de robots.

Tan, J. & Xi, N. (2004). Peer-to-peer model for the area coverage and cooperative control of mobile sensor networks. *In Proceedings of IEEE ICRA'04 Workshop on Environmental Robotics*.

Tews, A. & Wyeth, G. (2002). MAPS: A system for multi-agents coordination. *Adv. Robot (VSP/RSJ)*. 14(1), pp. 37–50.

Wang, P. (1989). Navigation strategies for multiple autonomous mobile robots. *In Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 486–493.

Yamashita, A. & Fukushi, M. & Ota, J. & Arai, T. & Asama, H. (2000). Motion planning for cooperative transportation of a large object by multiple mobile robots in a 3D environment. *In Proceedings of 2000 IEEE Int. Conf. on Robotics and Automation*, pp. 3144–3151.

Yamashita, A. & Ota, J. & Arai, T. & Asama, H. (2003). Motion planning of multiple mobile robots for cooperative manipulation and transportation. *IEEE Trans. Robot. Autom*. 19(2), pp. 223–237.

Yu, U. & Fukunaga, A. & Kahng, A. (1995). Cooperative mobile robotics: Antecedents and directions. *Technical Report* 950049.

Recent Advances in Multi Robot Systems

Edited by Aleksandar Lazinica

ISBN 978-3-902613-24-0
Hard cover, 326 pages
Publisher I-Tech Education and Publishing
Published online 01, May, 2008
Published in print edition May, 2008