A Hierarchical Hybrid Architecture for Mission-Oriented Robot Control

Manuel Muñoz, Eduardo Munera, J. Francisco Blanes, and Jose E. Simó

Institute of Control Systems and Industrial Computing,
Polytechnic City of Innovation,
Polytechnic University of Valencia, Spain
mmunoz@ai2.upv.es
www.ai2.upv.es

Abstract. In this work is presented a general architecture for a multi physical agent network system based on the coordination and the behaviour management. The system is organised in a hierarchical structure where are distinguished the individual agent actions and the collective ones linked to the whole agent network. Individual actions are also organised in a hybrid layered system that take advantages from reactive and deliberative control. Sensing system is involved as well in the behaviour architecture improving the information acquisition performance.

Keywords: Intelligent Robotics, Limited Resources Management, Embedded Systems.

1 Introduction

Some years ago, no one beyond the filmmakers imagined the current capabilities of the robots, spread from those who are cloistered in the factories, to the accompaniment and protocol Asian robots. Technology evolution in terms of computing power, sensory capability and mechanical improvements, allows to perform more autonomous robots, with complex tasks or behaviours, even in a heterogeneous collaborative mess. From this fact arises the need for control systems capable of handle robots while perform their tasks, and coordinate them to achieve cooperative tasks.

There is a lot of solutions to this problem, some focus on part of the problem, while others are dependent on a certain platform or architecture. The alternative proposal, mixes some of these solutions or paradigms and seeks to avoid some problems encountered by addressing the problem from a general point of view, and offering a platform independent solution.

From another point of view, these tasks require a lot of information from the environment in order to interact properly with it. Therefore in addition to the advanced sensing systems mounted on robots, special behaviour are required to encourage and/or optimise the acquisition of information. The proposed solution also addresses this idea.

M.A. Armada et al. (eds.), *ROBOT2013: First Iberian Robotics Conference*,
Advances in Intelligent Systems and Computing 252,
DOI: 10.1007/978-3-319-03413-3_26,

1.1 Previous Work

Robot control architectures have been a highly contested issue for years, and remains, in response to the large number of publications that keep coming on this subject. By hindsight, several trends can be observed: Deliberative Architectures, Behaviours-Based Architectures and hybrid solutions that mix both.

Early control architectures performed a deliberative execution because of its straight application despite of its requirement of a exhaustive previous knowledge of the situation to control. Nowadays this is still being used on some architectures like the one presented by Conor in [19], which is focused on the control of autonomous submarine vehicles.

The first steps in the model definition and formalisation of behaviour was exhibited by R. Brooks in [6]. The work describes an architecture for control mobile robots. Modelling some layers with different levels of competence. Each of these layers can access all the sensor data and generate control actions for the actuators. Separate tasks can suppress or inhibit inputs outputs. Thus, the lower layers can function as highly reactive mechanisms or reflections, while the upper layers are working to achieve the overall goal. Other classic behaviour based architectures like the proposed by and Arkin [3] remain strong influence over many current works like the presented by Cañas and Matellán in [7] where autonomous behaviours are generated by using dynamic hierarchies of small schemes. Thus, is obtained a scalable and extensible architecture which provide more flexibility than subsumption architecture and improves its performance in dynamic environments.

Some years later arises the trend of combine both previous paradigms. These kind of systems are called hybrid architectures or reactive-deliberative architectures. Stoytchev in [25] presents a hybrid robot architecture that combines three components: deliberative planning, reactive control, and motivational drives. Each of the three components addresses one of the challenges: To adapt quickly to changes in the environment; To understand high level human commands; and To be engaging and fun to use.

The amount of applications that requires a certain level of cooperation, and the increment of the capabilities of the agents involved, have promoted the number of researches focused in coordination architectures. Typical coordination system make use of a fusion of behaviours and an arbitration process in order to execute the most accurate action for each situation, a clear example was explained by Proetzsch in [21] where is proposed a solution called integrated behaviour-based control (IB2C). In this work is proposed the use of "behaviour modules" as the basic units of the architecture which contains information about the action, the rating, and the transfer functions of the behaviour. Those behaviours can be stimulated and inhibited in order to calculate the level of its relevance which in turns is reflected on the activity level of that behaviour to the current state. This activity level is used to perform a weighted fusion of the system behaviours for providing the output. One of the most relevant contributions of this work is the capability of implement different coordination methods separated in two main groups: arbitration or command fusion.

Actually most of the proposed solutions make use of fuzzy logic (FL) for the behaviour coordination. In [11] R. Huq offers a solution for behaviour-based control by combining the rigid state-based paradigm of the discrete events system (DES) with the flexibility of FL proposing the use of a fuzzy discrete event system (FDES) [15,22] as solution. The application of a DES ease the analysis of complex system that can be clearly described as a sequence of events. Transitions between states are usually based on sensory information, which implies that sensory uncertain can lead to an erroneous performance. Nevertheless, the application of FL can help to represent this uncertain, offering a mechanisms for dealing with that situations. In that architecture behaviours are defined by its associated action, the sensory information and a state-based modulation of its activity. That last element make use of fuzzy event matrices to generate predictions to estimate the activity of the behaviour. For the modelling of those matrix are used "fuzzy state vectors" which defines the probability of perform a transition between two states given a certain event. Based on the information contained on those vectors an arbitrator must determine the actions to perform. This paradigm is later extended on [12,13].

Although this, there are some coordination methods which offers some out the box solutions ,like the ones formulated on [16]. In this case is presented a multi-robot cooperation inspired in a biological system, a pack of wolves. That way is differentiated both alpha and betta roles which can be swapped between the pack members in function of its capabilities and collaborate to achieve a common goal, for the case, the simulation of an elk hunt.

The relevance of perceptual information for the execution of basic behaviours is clearly remarked in classical studies like Mataric in [17] where are remarked the advantages of centralised or distributed behaviours and its disadvantages, between some of them are related to the limitations in this time. Asama in [4] describes the importance of distributed information on robot systems making an analysis of functions distribution between the agents and path planning. On more actual works can be found some works where is implemented a communication between agents in order to share sensory information. A clear example of this can be found in [18] by Matellán where is compared three different fuzzy communication methods in order to share perceptual information about the environment and defines how this is reflected on its behaviours and the performance of its tasks.

1.2 Outline

As outline of this paper: After the introduction section 1.1 describes a review of related works explaining advantages and detected problems. A description of the used platforms and simulators is depicted in section 2. The proposed Mission Based System is shown in section 3. The group coordination approach is depicted in section 4. A clarify example is shown as a use case in section 5. A comment about the implementation issues is reported in section 6. The paper ends with some conclusions about the work in section 7.

2 Framework Overview

2.1 Development Platforms

The architecture introduced on this work is referred as a general solution for agents in networked environments. Nevertheless, all the proposed examples and validation tests will be focused on robotic platforms, including mobile and humanoid robots. Using mobile robots as agents of the system provides a reliable platform for validation, due to the wide range of sensors that can be equipped with, and its wheel-based locomotion. Humanoid robots on its side provides a more restricted platform, due to its complex locomotion an the limited number of sensors that can assemble. Validation tests have been performed on available robots and for which the research group has great experience in previous developments.

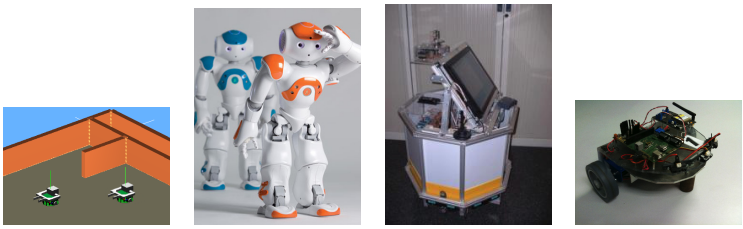
Mobile robots, just as has been described, are characterised by its displacement capabilities and its flexible configuration. Two different types of mobile robots are implied at this work, each one developed following its own philosophy. First is introduced the KertrolBot [26], as is showed on 1(d), a small size micro-controller-based robot which is propelled by a differential locomotion, is equipped with infrared sensors, and disposes of Wi-Fi communications. The goal of the KertrolBot is to provide solutions for small applications due to simplicity of its design and the low computational power.In the second case the Yet Another Intelligent Robot (YAIR), as can be observed in 1(c)[5], offers a similar locomotion but disposes of a bigger structure in which are integrated a screen, some more sensors as a camera or a laser range sensor, and the use of a computer as main operational unit. The YAIR offers a more wide range of applications because of its advanced capabilities and its computing power.

Giving way to the Humanoid Robot, the main challenge faced is dealing with biped walking. This locomotion is far more complex that the exposed on previous robots, and implies some restrictions on the mobility, the weight, and the stability, which limit the sensors that can be equipped with. In this particular case, we use the commercial robot Nao, displayed on 1(b) [23]. In spite of all the restrictions, this robot offers a good demonstrator for the development of some convoluted tasks like playing soccer.

For testing more complex environment situations, and disposing of more variety of robot models, we use a simulator, like Gazebo [14]. This one is a robot specific simulator which offers the capability of reproduce most of the real applications which are proposed on this paper, and enhance fast development and validation of the described architecture. An example of simulated robots can be observed in 1(a).

2.2 System Architecture

Beyond the execution of behaviours, agents must assemble mechanisms for performing some other type of jobs. Then, a scalable and flexible framework is established in order to group required tasks according to its functionality forming



(a) Simulator (b) Nao (c) YAIR (d) KertrolBot

Fig. 1. Presented Platforms

modules. These modules offer abstraction from their implementation providing an interface which can be easily accessed by other modules.

Perception Module. In almost every proposed situation, agents must obtain information about the physical world in order to perceive its environment. Depending on the environment and the possibilities, the appropriate sensor must be used. Thus the module must provide elaborated information as independent as possible from the source. Each one must be properly studied for adapting its raw sensory measures to a predefined data type. That adaptation process is the main objective of this module, which provides understandable information about the physical world to other modules that required it in order perform a proper operation. That is specially critical for the case of the Global Modelling and the Actuation modules.

Global Modelling Module. A reliability knowledge of the environment together with the agent's location in space brings the capability of use this information to take decisions about the best way to face its mission. That takes even more relevance when the agent have to perform some kind of path planing, or location tasks, in order to achieve its mission. To meet this need, current module offers a localisation method which allow to estimate the location of an agent in the space from the information provided by the perception module. But a model of the area is also needed in order to be able to discern its position in it. For that reason this module must also manage the environment map, generating it in case that no previous information is provided. Many techniques for both, localisation and environment mapping, are implemented as will be introduced on the Environment Interaction section.

Actuation Module. On the way to achieve its mission an agent may have to interact with the environment, which can involve from apply a control action to perform a displacement on the space. As will be introduced next, usually the actuation is raised as the result of a behaviour fusion, which reflects the desire of the agent to perform a certain action. Just as the opposite case of the adaptation phase described on the perception module, the main purpose of this

one is to translate the information provided by other modules, expressed in a defined data type, to a raw value required by the actuator that will depend of their characteristics.

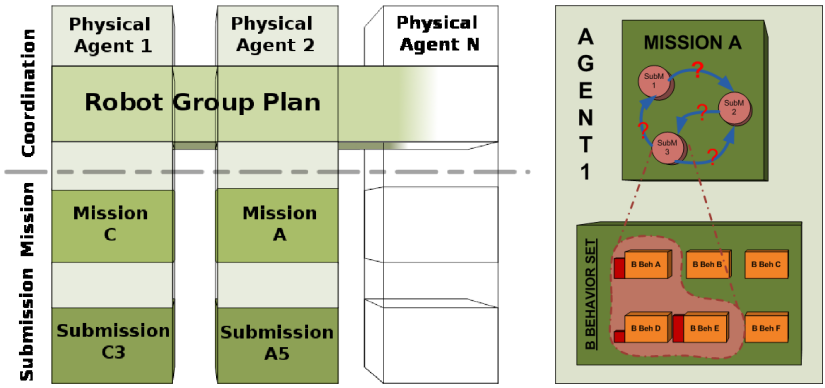
Communication Module. Some of the information managed in the previously mentioned modules, and specially the Perception Module, is sensitive to be shared with other agents of the system. In that way, sharing information between agents provides knowledge, including the state of the other agents, the progression of the mission and information about the environment. The establishment of this module enhances the coordination and brings the capability of setting up some levels of cooperation among robots, as well as the implementation of distributed consensus and data association mechanisms [1]. Those methods can be remarked as a hot topic according the amount of publications that refer to it. On next sections will be introduced how the proposed system offers a great base for integrating a distributed consensus layer.

Behaviours Module. The Behaviours module is responsible for generating commands. The work described in this paper mainly turns around this module. We propose an approach named "Hierarchical Hybrid Architecture for Mission-oriented Robot Control". System has been split hierarchically in two components:

- On one hand, the coordination side offers a high level of abstraction and manages all the information related to the collective strategy, consequently it can be represented as a module of distributed agents.
- On the other hand, the control side works with information from different levels of abstraction, but this information belongs to the individual behaviour of each agent. On Figure 2(a) can be noticed the general representation of a behaviour system.

3 Individual Behaviour – Mission/Submission System

The individual behaviour layer is responsible for executing the mission that has been assigned to that agent. An individual behaviour is defined as any behaviour carried out by one unique agent that leads its actions in a certain lapse of time. This behaviour is determined by a mission, which has been assigned to each agent from the upper layer as is show later on the section "Group Coordination". In most cases, individual behaviours can be composed by actions of heterogeneous complexity, for that reason this layer has been split in three different levels: Main behaviour, Mission, and Submission (or Mission State). On following sections will be described concepts ranging from the more basic levels to the mentioned Mission in ordered to ease its understanding. In Figure 2(b) is shown an overview of the interaction between this three levels.



(a) General Overview (b) Individual Overview

Fig. 2. Architecture Overviews

3.1 Basic Behaviour

A Basic Behaviour (or Basic Skill) is an action executed by an agent in response of stimulus, or precise commands transferred to a lower module to perform an action. Basic behaviours are strongly related to the agent performance in both input (perception capacity) and output (actuation capacity). It also is characterised by aiming to a simple and precise goal. A basic behaviour can not be divided and executed in a concurrent way.

Output Features: The output of a basic behaviour must actuate according the capabilities offered by the agent API, which is widely ranged between different kind of actions. In general case, it is defined as the establishment of a certain value as output information. On software agents this can be considered as a trigger between its own states. However, physical agents offers different options like, for example, establishing a voltage on an actuator, positioning a servomotor to a certain angle, making the final effector of a robot reach a position, or the establishment of a reference velocity for a mobile robot. It must be remarked that this performance is imposed by the kind of the current agent which is being working with.

Characterisation of Input Stimulus: Input signals that are used by the basic behaviours provides a low abstraction level. In most cases, this signals are used to close control loops which rule basic behaviours.

3.2 Behaviour Fusion

As has been explained on the basic behaviour section, basic skills must be simple and indivisible. One of the main reason for the establishment of that restrictions,

is to provide the system a mechanism to fuse those abilities, obtaining as a result the emergent behaviours that allows the resolution of more complex problems that can not be solved separately.

Motivational State: The concept of "motivation" is introduced for being used as a weighting element when a fusion of basic behaviours is performed. One behaviour, with a high level of motivation, will influence on the composition in a more representative way than other less motivated. In every behaviour, its motivation value, must be computed previously to the fusion. On this calculus may take part many factors as temporal and sensory data, the own logic state of the system, ifnextchar.etcetc..

Composition Function: The "composition function" is used in order to obtain the final value of the actuation for a certain variable based on the contributions brought by the different behaviours. As a basic composition function is proposed the weighted mean of the basic behaviours outputs, using the "motivation" as weighting element. Composition is calculated using equation 1. For the basic behaviour i, its motivation BM_i, is multiplied by its contribution BC_i. The result of the sum of all these products fro all the basic behaviours is normalised with the summation of the motivations. For that purpose can be used any other function, therefore the comparison of different functions, and the analysis of its influence on the output is relegated to a future work.

$$u() = \frac{\sum_{i=1}^{n} BM_i \cdot BC_i}{\sum_{i=1}^{n} BM_i}; \tag{1}$$

Disjoint Behaviours: Are called disjoint those behaviours that must/can not occur simultaneously because there is not physically possible, or because a design restriction. As more visual examples, when using biped robots as physical agents is not possible to perform a kick while the robot is walking, or in the case of wheeled differential robots is not possible to follow a straight line with a constant speed while is performing a turn on the vertical axis. For the establishment of the composition avoiding disjoint behaviours is proposed a modification of the previous Equation 1. The concept of composition table (equation 2) is introduced, in every row is indicated the composition coefficient of each behaviour with the rest. In this way, the element $c(i, j)$ shows the index of composition for the behaviour j with the i. The composition of a behaviour with itself must be 1, while the index with disjoints behaviours, which cant be executed at same time, must be 0. Therefore, this coefficient is dimensionless and could take real values between 0.0 and 1.0.

$$C_{i,j} = \begin{bmatrix} 1 & \cdots & [0.0 - 1.0] \\ \vdots & 1 & \vdots \\ [0.0 - 1.0] & \cdots & 1 \end{bmatrix} \tag{2}$$

The composition equation 1 has been modified for taking into account this composition matrix obtaining a new equation 3. The procedure to follow implies that composition coefficients must be extracted from the behaviour column with the highest motivation value. Thus new equation must be:

$$u() = \frac{\sum_{i=1}^{n} BM_i \cdot BC_i \cdot c_{i,j}}{\sum_{i=1}^{n} BM_i \cdot c_{i,j}};\ j = \text{more motivated behaviour index.} \quad (3)$$

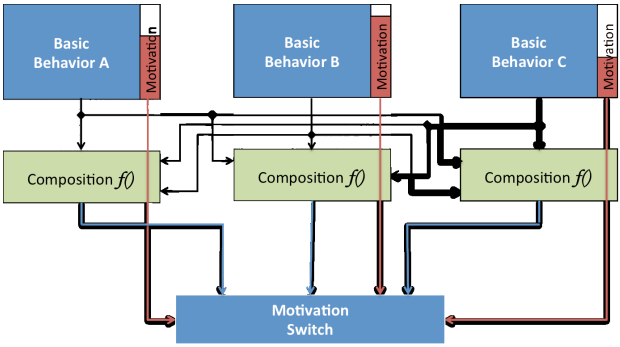


Fig. 3. The fusion of Behaviours A, B and C are conditioned by their levels of motivation

Fatigue Factor: The fatigue factor weights the contribution of a behaviour depending on the lapsed time from its activation. The contribution values are ranged from 0.0 to 1.0, and are obtained as the product of other 2 subfactors:

- Rise subfactor: Favours the progressive introduction of the behaviour. Its value is initially 0.0 and is linearly increasing until reaches the rising time when must be 1.0.
- Fall subfactor: This factor limits the action of a behaviour if is reached the fatigue time and remains active because has not accomplished its goal. Its initial value is 1.0 and is decreasing until 0.0. After the falling time the values is maintained till expires the blocking time.

These factors are calculated from 3 parameters, rising time, fatigue time and blocking time. This parameters will be established according to the dynamic of each behaviour.

3.3 Submission or Mission State

A submission is composed by a set of the available basic behaviours. When a state is active every behaviour of the set is also activated. The activation of several behaviours leads to the emergence of new behaviours as a result of the fusion of its control actions. A submission is also characterised by its table of composition and all the basic behaviours contained.

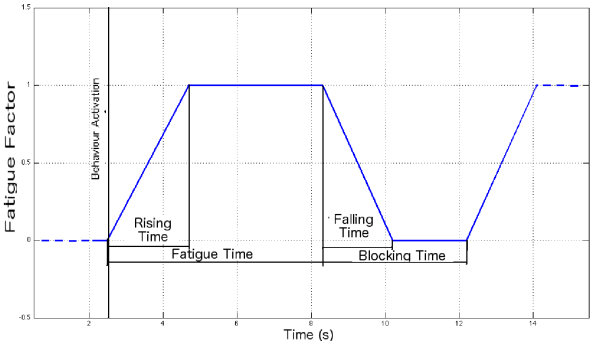


Fig. 4. The application of the Fatigue factor avoids to get stuck on the execution of non-progressing behaviours

3.4 Mission

Is defined as a "mission" the complex task realised by an agent, as a part of a group, in order to reach a certain objective assigned for the upper coordination level. Missions may have a concrete goal or may execute a repetitive task. If an agent is performing alone, the mission to achieve will not be modified along time. Missions are represented by a HFSM (Hierarchical Finite State Machine), in which graph are depicted two types of elements: states and transitions. States are explained in detail on the "Submission or Mission State" section. One of the states will be marked as the initial state and will be the first to activate once the mission starts. Transitions between states are performed after the evaluation of a set of sensory conditions, and can be traversed in one direction only. Those conditions ensures that the partial objectives of the state has been successfully accomplished. It can be considered that the mission does not provide an output for itself, but takes the output of the active state in the moment of the execution. Mission requirements will evaluate the state of accomplishment of the submission in execution. It must be to distinguish two cases:

- Global requirements: will be the sum of the requirements of every state in the mission, along with the requirements of every existing transition.
- Actual requirements: are the requirements of the actual states, with the sum of the requirements of its exit transitions.

3.5 No Goal Oriented Behaviours

Although the proposed architecture aims to the accomplishment of goals, not all the behaviours involved are directly related to the objective. The main functionality of these behaviours is to provide the needed support for assure that the goal-oriented behaviours dispose of all the requirements for its proper execution. Between these requirements most usually can be found the need of an environment interaction or the knowledge of the status of the other agents in

the network. Therefore, this section will fall on the characteristics of the behaviours employed for covering this demand and the way that is managed in the architecture.

Environment Modelling and Localisation: Most of the executed tasks that are relevant to the coordination system, are also related with the interaction with the environment. That is specially critical when the agents have to perform some kind of path planing or location tasks in order to achieve its objectives. For providing them the knowledge of the characteristics of the environment in which are located can be provisioned a previously generated map, or as alternative can make use of simultaneous localisation and mapping (SLAM) [8] techniques that allow the agents to generate a map of its initially unknown surrounding thanks to the sensory information.

Typically the use of mobile robots as agents implies the need of being able to locate its own position in the map for developing its tasks in an proper way. Many techniques have been used for that purpose, like the Montecarlo Particle Filter [9] Monte-Carlo Particle Filter (MCL) and the some variants of the Kalman Filter (KF) as the Unscented Kalman Filter (UKF) [24], although it also have been developed some derived techniques by taking profit of the advantages of both them. That way in [20] is presented a location solution for humanoid robots, in a soccer game environment, which propose a combination of these techniques.

A good knowledge of the environment and their location in space brings the possibility of using this information to take decisions about the progress of the mission. The exchange of data between agents allows to perform a better distribution of the tasks, enhance the environment model, or detect dynamic obstacles. That way, next section will describe the advantages provided by the data distribution between agents and the establishment of consensus procedures.

Cooperation and Consensus: In a network of agents, where a common objective is fixed, the fact that every agent is working on its own without caring about the state of other ones does not take profit of the advantages of these systems. Any kind of cooperation between agents will provide a remarkable enhancement for mission execution.

One of the tasks that will be improved by this cooperation is the environment modelling. If a map of the work area is provided the detection of dynamic elements is a simple decision, but it is not also on those situations where agents must generate its own map. Once an element of a new area are spotted there is no initial distinction about if is an static or a dynamic obstacle. This situation can be solved by a consensus phase where agents which also has been explored this area contribute with their information, as can be seen on [2].

Some of the most common application of cooperation to the environment modelling are intended to perform efficient area exploration. This kind of cooperation are widely used in mobile robot and Unmanned Aerial Vehicles (UAV) networks [10]. An exploration performed with several robots which provides a certain overlap between the sensed range allow a fast association between the distributed information an provides an agile method for environment modelling.

Focusing on the mission development, cooperation is also a clear benefit as it improves the performance and speeds up the execution. In a network of agents which does not perform any type of cooperation, tasks must be preassigned independently of its state. By enabling a consensus phase, this state, which characterises the agent (position, tasks in execution, ifnextchar.etcetc.) may be considered in order to decide which are the most qualified robot to execute a new task when its required.

Task Integration Approach: The need of localisation implies the execution of several tasks in order to guarantee a accurate estimation. That way several behaviours have to be exclusively or tightly related with that kind of tasks. As any other behaviour active they must also be fused in addition to other behaviours related purely related with the mission execution. In most of the cases required actions are intended to the sense of its surrounding, but in some other cases implies active motion in order to perform a pursuit of localisation marks.

Examples: Following the example explained in previous sections. Physical agents need to be localised and with a good sensory feedback. Thus additionally to the 2 Basic Behaviours described in section 5 a Non Goal Oriented Behaviour is joined. The behaviour is Landmark Following:

- Landmark Following.
  - The objective of the physical agent is to trace its trajectories as near as possible to the available landmarks. That way, positioning is not lost.
  - The action, output, or control action, taking into account the exposed on previous examples, can be one of the components of the velocity vector which guide the agent near to landmarks.
  - Perceptual requirements are the position of the surrounding landmarks.

4 Group Coordination

The lower layer need the indication of what mission must be executed, for that reason is implemented a group coordination layer. By the application of that layer can also be obtained some other additional benefits, thanks to a good coordination between agents any activity can be performed in a more efficient way than in a individual approximation. Next is introduced the different sublevels in which is decomposed this layer.

4.1 Tactics

Tactics will be defined as a set of missions to achieve by a group of agents. None tactic is preassigned to a mission nor an agent, but must establish certain restrictions referred to the minimum resource types that must have an agent to

perform a concrete task. The number of agents, which realises this tasks, does not have to be known a priori, therefore it must be established a priority task assignation criteria. In the same way, it is possible to exist duplicates (an agent perform more than one) according to the available resources.

Tactic Execution. Mission Assignation: The most adequate way for a proper execution, is the assignment of the different missions to each agent in order to reach the goal of the strategy. Each mission is assigned in a reactive way attending to its restrictions and the capacities of the agents. The assignation criteria can also depend on the sensitisation, or attend to an opportunistic pattern. Tactic objectives are the same as the strategy ones, but additionally must optimise the distribution of missions between agents.

4.2 Plan

The Plan describes the set of actions that the group of agents must perform for reaching the goal. This actions are grouped in Missions and this in turn are grouped in Tactics. This plan aims to the achievement of set of actions, Tactics and/or Missions, executed in a determined order, and carried out by a group of agents which can swap theirs missions in a opportunistic way. An HFSM describes the Plan in which each state is a Mission.

5 Experimental Use Case

This example tries to clarify the concepts explained above. The context for the example is the scenario shown in Figure 5. There a factory is simulated where the handling of materials is performed by 4 physical agents (gripper differential robots) that transport the freight from a charging dock and to the corresponding download box.

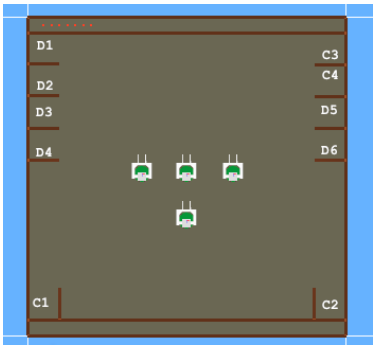


Fig. 5. Team Behaviour Example

5.1 Robots Coordination: Plan

In the coordination level is defined a three state plan and the corresponding tactic for each of these states, each one of this named as tactic.

1. **Merchandise reception:** A big amount of freight is receipt.
2. **Standard merchandise movement:** Some freight is receipt.
3. **Merchandise dispatch:** No freight is receipt.

5.2 Tactic

In the case that a big amount of freight is receipt a state machine evolves to the "Merchandise reception" state. The wheelbarrows must unload the freight and take the charge to one of the unload places. The tactic consists of 4 Missions, but these are not assigned in principle to any particular agent.

- **Mission I:** Freight Reception.
- **Mission II:** Freight Reception.
- **Mission III:** Freight Reception.
- **Mission IV:** Battery charging.

Tactic Execution. Mission Assignation: This scenario use 4 robots to achieve the goal. The tactic sets 3 equal missions to receipt the freight and leaving it in the proper place, while 1 robot reload its batteries. The assignment is done according to battery load criteria.

5.3 Mission

As example of Mission "Freight Reception" is explained: The goal of this mission is collect freight from a charging dock and unload it at the appropriate place. Another mission can be, relocating objects: load collection in a given box to move it to another.

5.4 SubMission

The first mission described above, could consist of a set of submissions as follows:

- Go to the loading dock.
- Adjust the freight to be picked up.
- Loading the freight.
- Go out of loading dock.
- Go to the unload site.
- Download freight.
- Go out of unloading dock.

As this is a simple example, all submissions are executed sequentially and in loop, but not always be this way.

5.5 Basic Behaviour

As example the first submission is explained. This submission is formed by 2 Basic Behaviour: *Displacement to site* and *Environment object collision avoiding.*

Displacement to site:

- The objective of the physical mobile agent is to perform a movement in the direction of a interest goal.
- The action, output, or control action, as the shown on the previous example, relies on the performance of the agent. In addition, the behaviour must obtain the direction and velocity of the displacement required for reaching the position of the interest place.
- The relative position between the robot and the goal site are the perceptual requirements.

Environment object collision avoiding:

- The objective of the physical agent is to avoid to collide with its surrounding objects while is in movement.
- The action, output, or control action, taking into account the exposed on previous examples, can be one of the components of the velocity vector which avoids the collision with a certain object.
- Perceptual requirements are the position of the surrounding objects.

6 Implementation

Although the model referred along this document is functional and has been tested, the evolution, tests and improvements are always progressing. Current implementation has been developed on C++, using a computer with a Linux distribution as main operative system, and has been tested using both a simulator which offers virtual representation of robots, and real mobile robots as elements to control. The robots used in this test are the humanoid platform Nao. In both cases tests have been done using the framework developed to attend the Standard Platform League (SPL) in the RoboCup[1] championship by the Hidalgos Team.

6.1 Development Model

The behaviour module has been implemented with the aim of be usable over several robots, so has been designed robot-API independent. Different behaviour components (Basic Behaviours, Missions, Submissions, Plans, Tactics, etc) use the c++ heritage mechanism to take the architecture benefits. Thus, one base class is generated by each component in a header file. The behaviour designer just must generate a class per component that extend the appropriate basis class, and add the behavioural functionality code. Next subsections explain each component in detail.

[1] `http://www.tzi.de/spl`

Basic Behaviour. In a Basic Behaviour derived class, the functionality has been split into 3 functions/blocks:

- Motivation Update: in this function user has to calculate and store the value of the current motivation for the behaviour in edition.
- Contribution Update: in this function user must set a proper values for contributions over which it has influence. Different behaviours could act over different actuators or system variables.
- Actuation function: takes the result of the fusion process and perform the actuation. Each behaviour decides which contributions need, and how to use them to actuate on the robot API.

Submissions. As was described, a submission is a set of Behaviours. In the constructor the designer must add all the required basic behaviours. Additionally user must afford a function that calculates and provide the accomplishment level of the current submission.

Missions. A Mission is a set of submissions that are executed following the criteria imposed by a finite state machine. That way, the programmer has to specify states and transitions to depict the flow of the machine.

- The states are conformed by submissions, the designer just need to instantiating the previously defined.
- Transitions are needed to indicate the relationships between states. The code that manage the transit condition, and the states of input and output, define the transition.

6.2 Behaviour Migration

After the design of behaviours, the code is compiled and linked as a dynamic library. The library could contain the complete behaviour or a subset. Availing of this mechanism, is possible to modify part of the behaviour (e.g. basic behaviour), link alone in a new library and send it to the robot, then the module is requested to load this behaviour in place of the current one.

7 Conclusions

The presented hybrid architecture presented in this paper take advantages from reactive and deliberative control. The aim of this architecture is to manage the behaviour of a heterogeneous group of physic agents. The layered structure allows a easy behaviour definition, obtaining a good reactive response in the low level and a simple planning in the deliberative level.

The group coordination level, deal with the missions assignment according the robots state and resource availability.

Finally architecture introduces sensory tasks in the behaviours system, thus resultant actions could improve the perceptual performance. System is being tested in a simulator and different mobile robots.

Acknowledgments. This work has been partially supported by the Spanish Ministry of Economy and Competitiveness under the CICYT project Mission Based Control (COBAMI): DPI2011-28507-C02-02, under coordinated project High Integrity Partitioned Embedded Systems (HI-PartES): TIN2011-28567-C03-03, and under the collaborative research project supported by the European Union MultiPARTES Project: FP7-ICT 287702. 2011-14.

References

1. Aragues, R.: Consistent data association in multi-robot systems with limited communications. Robotics: Science and Systems, 97–104 (2010)
2. Aragues, R., Cortes, J., Sagues, C.: Distributed consensus on robot networks for dynamically merging feature-based maps. IEEE Transactions on Robotics (2012)
3. Arkin, R.C.: Motor schema based mobile robot navigation. The International Journal of Robotics Research 8(4), 92–112 (1989)
4. Asama, H., Habib, M.K., Endo, I., Ozaki, K., Matsumoto, A., Ishida, Y.: Functional distribution among multiple mobile robots in an autonomous and decentralized robot system. In: Proceedings of the 1991 IEEE International Conference on Robotics and Automation. IEEE (1991)
5. Benet, G., Blanes, F., Martínez, M., Simó, J.: A multisensor robot distributed architecture. In: IFAC Conference INCOM 1998 (1998)
6. Brooks, R.: A robust layered control system for a mobile robot. IEEE Journal of Robotics and Automation 2(1), 14–23 (1986)
7. Canas, J.M., Matellán, V.: Dynamic schema hierarchies for an autonomous robot. In: Garijo, F.J., Riquelme, J.-C., Toro, M. (eds.) IBERAMIA 2002. LNCS (LNAI), vol. 2527, pp. 903–912. Springer, Heidelberg (2002)
8. Choset, H., Nagatani, K.: Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization. IEEE Transactions on Robotics and Automation 17(2), 125–137 (2001)
9. Fox, D., Burgard, W., Dellaert, F., Thrun, S.: Monte carlo localization: Efficient position estimation for mobile robots. American Association for Artificial Intelligence, 343–349 (1999)
10. Hu, J., Xie, L., Xu, J.: Vision-based multi-agent cooperative target search. In: Control Automation Robotics & Vision (ICARCV), pp. 895–900 (2012)
11. Huq, R., Mann, G.K.I., Gosine, R.G.: Behavior-modulation technique in mobile robotics using fuzzy discrete event system. IEEE Transactions on Robotics 22(5), 903–916 (2006)
12. Jayasiri, A., Mann, G., Gosine, R.G.: Mobile robot behavior coordination using supervisory control of fuzzy discrete event systems. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 690–695 (2009)
13. Jayasiri, A., Mann, G.K.I., Gosine, R.G.: Behavior coordination of mobile robotics using supervisory control of fuzzy discrete event systems. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 41(5), 1224–1238 (2011)
14. Koenig, N., Howard, A.: Gazebo-3d multiple robot simulator with dynamics. Technical report (2006)
15. Lin, F., Ying, H.: Modeling and control of fuzzy discrete event systems. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 32(4), 408–415 (2002)

16. Madden, J.D.: Multi-robot system based on model of wolf hunting behavior to emulate wolf and elk interactions. In: 2010 IEEE International Conference on Robotics and Biomimetics, ROBIO, pp. 1043–1050 (2010)
17. Mataric, M.J.: Interaction and intelligent behavior. Technical report, DTIC Document (1994)
18. Olivera, V.M., Molina, J.M., Sommaruga, L., et al.: Fuzzy cooperation of autonomous robots. In: Fourth International System on Intelligent Robotics Systems, Lisboa, Portugal (1996)
19. McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., McEwen, R.: A deliberative architecture for auv control. In: IEEE International Conference on Robotics and Automation, ICRA 2008, pp. 1049–1054. IEEE (2008)
20. Munera, E., Muñoz, M., Simó, J., Blanes, F.: Humanoid Robot Self-Location In SPL League. In: Comité Español de Automática (CEA), XXXIII Jornadas de Automatica, 797–804 (2012)
21. Proetzsch, M., Luksch, T., Berns, K.: Development of complex robotic systems using the behavior-based control architecture iB2C. Robotics and Autonomous Systems 58(1), 46–67 (2010)
22. Qiu, D.: Supervisory control of fuzzy discrete event systems: a formal approach. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 35(1), 72–88 (2005)
23. Aladebaran Robotics. NAO Software Documentation 1.12. Technical report (2012)
24. St-Pierre, M., Gingras, D.: Comparison between the unscented Kalman filter and the extended Kalman filter for the position estimation module of an integrated navigation information system. In: 2004 IEEE Intelligent Vehicles Symposium, pp. 831–835 (2004)
25. Stoytchev, A., Arkin, R.C.: Combining deliberation, reactivity, and motivation in the context of a behavior-based robot architecture. In: Proceedings of the 2001 IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp. 290–295 (2001)
26. Nicolau, V., Muñoz, M., Simó, J.: KertrolBot Platform. SiDiReLi: Distributed System with Limited Resources. Technical report, Institute of Control Systems and Industrial Computing - Polytechnic University of Valencia, Valencia, Spain (2011)