Università degli Studi di Napoli Federico II

Facoltà di Ingegneria
Dipartimento di Informatica e Sistemistica

Tesi di Dottorato di Ricerca in Ingegneria Informatica ed Automatica
Coordinatore: Prof. Francesco Garofalo
November 2011

Attentional Mechanism for Sensory-motor Coordination in Behavior-based Robotic Systems

by

Mariacarla Staffa

Thesis Supervisor: **Prof. Bruno Siciliano**

Thesis Co-Supervisor: **Prof. Ernesto Burattini**

Submitted to the Faculty of Engineering, University of Naples Federico II, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Printed in Italy.
Naples, November 2011.

Contents

Abstract

This thesis focuses on the problem of efficiently allocating resources for enhancing the performance of an autonomous robotic agent. Such an agent is expected to operate in complex dynamic environments by continuously monitoring its internal states and the external events.

These requirements give raise to countless problems that have populated research in the autonomous robotics community in the last two decades. Among these issues, one of the most relevant is to coordinate different low and high-level behaviors, giving them, from time to time, different priority values both for resource allocation and for action selection processes. The main problem in achieving this requirement is that the number and complexity of the stimuli received by each behavior may be quite high and also the effects on the emerging activity may be very hard to foresee.

It is clear that it is not possible for the robotic control system to process all the incoming information, especially for real-time applications. Thus, it becomes necessary to build mechanisms able to guide this sensory input selection process and to choose the best action to perform, assuring an efficient use of the robot limited sensorial and cognitive resources.

For this purpose, attentional mechanisms, balancing sensors elaboration and actions execution, can be very useful since they play two main roles: they focus the attention on salient regions of the space and they distribute resources and activities in time.

As a result of the application of these mechanisms within the robotic control system for sensory-motor coordination, the robot behavior is improved: the robot becomes able to react faster to task-related or safety-critical stimuli and to opportunely split resources among concurrent behaviors.

Attentional mechanisms applied to autonomous robotic systems have already been proposed, but mainly for vision-based robotics; conversely, the contribution introduced by the present thesis is the use of an artificial attentional mechanism suitable both for optimizing the use of resources and for execution monitoring and control.

Part I

Prologue

Chapter 1

Introduction

1.1 Motivations

During the last years we have witnessed the construction of about one million of functional robots in the world, the most coming from the industrial robotics [1]. Although the introduction of mobile robots in everyday life is one of the most discussed topic nowadays, are still a few the mobile platforms developed and actually in use, above all when compared to the number of robotic arms successfully used in the manufacturing industry already since long time. Hence, despite exceptions such as the automatic cleaning machines operating in the Paris metro [2] or the famous iRobot robots for home cleaning [3], in comparison to the evident success of robots in industrial scenarios, mobile robots still have a marginal relevance in our society. This is probably due to the fact that extending the usage of robots to homes, offices, hospitals and public places, in general, represents an extremely challenging step that state-of-the-art research has not fully solved yet. In order to achieve such a level of integration of these mobile robots in our society, a certain number of issues must be addressed; among all, the most relevant seems to be "how to make a robot fully autonomous".

The abilities which make a robot autonomous are many and varied. These depend not only on the physical characteristics of the robot (memory limits, number and type of available sensors, sensor accuracy, computing power, etc..), but also on the domain and purpose of the robot. In general, however, we can say that

one of the main requirements imposed by the design of this kind of autonomous systems is the capability of operating in real time in a complex dynamic environment. An autonomous agent achieve this goal by continuously monitoring the internal processes and the external environment and by coordinating different low-level activities (such as obstacles avoidance [4], walls follow, gates crossing, etc.) with high-level strategies (such as recognizing people [5], planning complex tasks [6], etc.), giving them different priority values, depending on the possibility that a collision will occur, or that some parameter is minimized, and so on. The low-level activities are closely related to vehicle control and safety. They can be realized by applying the principles of a purely reactive architecture [7]. The high-level tasks are, however, generally produced by more complex processes (such as route planning or the calculation of the next movement according to an optimal trajectory [8], [9]), which often involve the need of some internal representation model of the surrounding environment [10]. These types of activities have, therefore, high computational costs both for the elaboration of data coming from sensors and for the acquisition of knowledge about the environment and the consequent updating of the world representation model.

These considerations lead to the conclusion that dealing with the real world using physical devices cannot be accomplished without explicitly considering the need to find a mechanism to minimize the computational costs arising from all these activities in order to improve the performance of the robotic system. This also because our future perspective is to create systems that will be able to interact with humans in their domestic environment. These systems can hence be characterized by low computational capacity with respect for example to industrial systems, but they should be embedded on a mobile device of limited size, so that any efforts will then be produced forward that direction.

The starting point of this thesis is that, if we want to consider the construction of autonomous agents working in these highly complex environments, both the most efficient use of limited sensorial and cognitive resources, and a mechanism for coordinating the concurrent behaviors, involved in achieving a global emergent goal, must be assured.

In psychology and neuroscience, similar capabilities, mediated by frontal areas of the brain [11], are called executive functions [12]. Executive functions allow

orchestrating cognitive and automatic processes providing coherent, flexible and adaptive behaviors. In this coordination process, also called cognitive control, attentional mechanisms have a central importance [13]. Beyond their role in orienting perception by focusing on relevant stimuli, attentional mechanisms are considered as key mechanisms in action control, and in particular, for tasks involving planning and decision making, managing dangerous or new situations, and habitual responses to inhibit [14].

Indeed, attentional mechanisms play a crucial role in cognitive control [13] and sensory-motor coordination, since they are able to manage sensor elaboration, by consequently affecting the action execution. Let us consider that in our daily life we always have to perform actions consistent with our goals and to do so we must choose, among the available information, those profits, by separating them from the others. We use our attentional mechanism for this purpose. Therefore the choice of the relevant information is affected by the action goal [15]. Hence, due to the variety and complexity of real environments, also an autonomous robot needs an *attentional system* or an *executive control* able to ensure an efficient use of its limited resources and to balance sensor elaboration and action execution.

Inspired by human attention mechanisms, the key idea has been to provide a robotic system with an attentive executive controller [16] in order to regulate the sensory-motor coordination. We implemented the attentional mechanisms in both the direction of "selective" and "divided attention" with the aim to achieve two main goals. On the one hand, by means of the selective attention mechanism, the agent can direct sensors towards salient sources of information filtering the available sensory data, and manage sensors processing in an efficient way (since focusing on relevant information prevents unnecessary information processing, and consequently it reduces the computational load due to non-salient sensory data elaboration). On the other hand, the mechanism of divided attention can be used to split resources among different concurrent tasks, by giving them different priority levels, coordinating in this way multi-task activities. The system is based on the concept that attentional mechanism lies perception and action.

We proposed simple mechanisms for scheduling the attention based on an adaptive modulation of sensor sampling and action activation. In particular, we introduced a behavior-based architecture where each behavior is endowed with an

internal adaptive clock, that can speed up or slow down the sensor reading frequencies according to both the robot-environment interactions (bottom-up influences) and the internal processes (top-down influences). The process of changing the frequency of sensory readings is interpreted as an increase or decrease of attention towards relevant behaviors and particular aspects of the external environment in a way that the higher the frequency, the higher the resolution at which a process is monitored and controlled. We provided a kind of supervisory attentional system (Norman and Shallice like [14]) based on the selective attention mechanism able to filtering the available data, regulating sensing rate and consequently the resulting action activations. Then we extended this architecture by introducing some particular coordinating mechanism based on mutual influence rules implementing the divided attention capability with the aim to monitor and regulate multiple concurrent behaviors. The greatest challenge in constructing this type of attentive systems is to balance the benefits arising from the use of this attentional mechanisms (such as the efficient use of resources) and the risk of inaccurate information from the environment (losing the accuracy of the system). In fact, by adopting large latencies for sensor sampling, we certainly have an improvement in performance, but choosing much too large latencies, we could make the robot unsafe since the environment might change too much in between two consecutive readings. For this purpose we introduced suitable monitoring strategies, and as natural extension of the system, we adopted some learning techniques, based on an evolutionary approach, in order to learn the key parameters regulating these strategies, improving in this way the robot skills in dealing with such an unpredictable and dynamic environment.

Finally, in order to build embedded systems, we then implemented this same attentional control system through the formalism of neural networks, whose transfer on FPGA devices seems to be more immediate.

1.2 State of the Art

In this section we will provide a brief survey on the attentional systems topic. We will outline the most relevant works since regarding attentional systems in living beings we can find a very extensive literature. One of the reference books

on the subject is [17] which describes the psychology of attention and gives some details on many psychological attention models, such as the visual search models accurately treated by [18], [19]. A survey on computational attention systems that aims at bridging the gap between the research on human and computational visual attention can be found in [20]. There are also some recent contributions [21], [22], from which it appears that the attentional systems are broadly divided into two strategies based on the sensorial data flow: the bottom-up and top-down attentional strategies [23]. The bottom-up strategy takes into account the salient physical stimuli and sometimes is reported as a "pre-attentional stage", because in living beings it modulates the activity of certain pre-specified and task-specific detectors. The bottom-up strategy is well described in [24], where the authors present a bio-inspired model for determining interesting positions on the visual field based on some particular color, its brightness and on edge orientation maps. Strategies based on top-down information streams are presented in [25], in which they are related to high level task information that guides the search process to regions in which the goal objects are more likely to be found. The determination of the top-down-cues comes from higher brain areas like knowledge, motivations and emotions. One of the first computational models of visual attention was introduced by Koch and Ullman in 1985 with a detailed description of the winner-take-all approach [26]. Recently, several research groups have used information-theoretic approaches to determine visual saliency [27]. The latter also tackles the aspect of top-down saliency for object recognition by determining salient features that best distinguish a visual class from other classes [28]. Top-down information in the form of knowledge about the scene and its visual layout was used by Torralba et al. to guide visual attention to relevant parts of an image [29]. Then, it has been shown that the interaction of bottom-up sensory information and top-down attentional influences creates an integrated saliency map, that is, a topographic representation of relative stimulus strength and behavioral relevance across visual space. Many researchers have worked on this topic proposing simple frameworks to think about how salience may be computed in biological brains [30],[26],[24], [31].

In conclusion, attention-based control is an emerging issue, in particular for vision-guided mobile robots. Several approaches in literature address the problem

of feature extraction to support task execution [32], localization, mapping, and navigation [33, 20, 34]. For instance, in [32] an attentional behavior is learned by pairing actions and image features. There is also some bio-inspired work [35], where the authors implemented an integrated neural architecture, modeled on human executive attention, which was used to control both reactive and willed action selection. But their approach has been tested only in simulated robotic agents and it focused on an attention-based learning mechanism, extending the Norman and Shallice model. Then some authors applied some implementation of the visual SLAM based on attentional landmarks for robotic applications [36] or a bio-inspired robotic system for localization [37]. Both used salient objects in the environment as navigation landmarks, thus used the attentional mechanisms inspiration just for improving the robot visual capabilities.

Mechanisms for executive and divided attention in robot execution monitoring are less explored. In [38], the authors investigated executive attention in mobile robotics tasks proposing the deployment of a supervisory attentional system inspired by [14]. Concurrent tasks interacting with the attentional processes have been considered in [39] where we find a robot architecture integrating active vision and tasks execution. However, here divided attention is not considered while attentional and goal-directed behaviors are integrated and coordinated using a perceptual memory.

1.3 Contributions of this Thesis

The contribution of this thesis is that, differently from other approaches presented above, we are interested in artificial attentional processes suitable not only for directing attention towards salient source of information, but also for the executive control. In addition, we want to apply our architecture to real devices dealing with problems of limited resources. We know that the robotic platforms have limited computational power similarly to the physical constraints of humans: at one point in time, they can only go toward a particular location, choose one interesting object, interact with an operator and grasp one or a few objects. Thus, a mechanism that selects the relevant parts of the sensory input and decides what to do next is essential. And since a real robot has to operate in the same environments as

humans, it is reasonable to imitate the human attention system to fulfill these tasks [20] as it is reasonable to assume that attentional mechanisms have evolved to fulfill certain other functions like *Selection-for-action* [40].

We start from the consideration that behavioral studies on normal and brain-damaged individuals provide convincing evidence that attentional mechanisms link perception and action [41]. Indeed the perception of objects results in the generation of both visual and motor signals in the brain, irrespective of whether there is an intention to act upon the object. Many authors demonstrate that shifts of attention to the location of visual objects automatically generate some motor response codes ([42], [43], [44], [45]). Hence, the perceived objects automatically generate motor codes based on the actions most highly associated with them ([46], [47]). Coordinating movements basing on the salient input signals is advantageous in terms of both execution time and efficiency. For example it permits regulating the gait while in motion to avoid an obstacle.

In this direction, we propose simple mechanisms for scheduling the attention based on an adaptive modulation of sensors sampling and action activations.

Our main source of inspiration comes from the supervisory attentional system proposed by Norman and Shallice [14]. This system is able to suitably combine deliberative and reactive activities, and to monitor and regulate multi-behavior robotic system as in Khaneman [48]. More precisely, the Norman and Shallices model [14], [50], consists in two processes operating in the selection and control of action: (1) a contention scheduling, dealing with well-learned sequences of behaviors and (2) a supervisory attentional system (SAS), allowing for conscious control of behavior. This model of action control is supposed to be intimately associated with the goal pursuit [51]. The authors assume that well-learned action sequences are represented in schemas, and that these schemas may be activated by appropriate cues (either internal or external). When only one schema is activated, this schema controls behavior. When multiple schemas are activated, a selection process selects the one with the highest level of activation (the activation level of a schema is determined by the cues and context, as well as by processes of lateral activation and inhibition). Then in cases where conflicting schemas, or in novel tasks, the SAS comes into play. The SAS provides attentional, conscious control over behavior by changing the activation levels of different schemas, thus creating

novel and adaptive sequences of behaviors.

Starting from this model we designed a robotic control system based on the attentional mechanisms, able to activate the suitable behavior with respect to the environmental circumstances and to the robot internal motivations. In particular, the choice of the behavior to be activated depends on its activation frequency. This frequency value represents in some way the amount of attention focused on that particular behavior and it depends, as well as in the Shallice's model, on both internal and externals cues or motivations. In order to manage potentially conflicting behaviors, we adopted some mutual influence rules able to regulate dependencies among concurrent and conflicting behaviors. Finally, we added to this schema a supervisory attentional control able to adaptively re-schedule and re-planning activities, in case of new or unexpected events.

We started from the problem concerning the efficient allocation of resources in order to enhance the performance of an autonomous robotic agent, and we addressed this problem by means of the adoption of the system realized, since it permits:

- efficiently spending resources to monitor the surrounding environment and the internal processes — restricting these processes to a limited subset of sensory data enables efficient processing;
- adapting to environmental changes by reacting faster to task-related or safety-critical stimuli;
- coordinating active sensing and control strategies;
- harmonizing multiple behaviors while maintaining the features of adaptability and reactivity.

Closely related to our system, in [52] Stoytchev and Arkin proposed a hybrid architecture combining deliberative planning, reactive control, and motivational drives. In this context, the internal state was represented by motivational variables affecting action and perception. Analogously to our framework, periodic activations of behaviors as circadian rhythms and time-dependent motivational processes were deployed; however, here internal clocks were not directly used for attention selection and behavior modulation.

Other authors dealt with flexible/adaptive behavior realized through timed activations. For example, [53] presented a parallel architecture focused on the concept of activity level of each schema which determines the priority of its thread of execution. A more active perceptual schema can process the visual input more quickly and a more active motor schema can send more commands to the motor controller. However, while in our approach such effects are obtained through periodic activation of behaviors, in [53] the variables are elaborated through a fuzzy based command fusion mechanism.

Our attentional sampling can be also related to flexible scheduling for periodic tasks in real-time systems [54, 55]. Here, as in our system, a periodic modulation is exploited to degrade computation and keep balanced the system load. For example in [54], the authors propose an elastic model to decide how to change the sampling period associated with a task. The model works for increasing the period of different jobs any time there is a significant variation in one job. Moreover, whenever a periodic task terminates or decreases its rate, all the tasks that have been previously modified can increase their utilization or return to their nominal periods, depending on the amount of released bandwidth. Similar techniques can be incorporated in our framework; however, in our case sampling rate depends not only on the computational load, but also on the salience due to environmental changes, motivations, and goals.

1.4 Thesis Outline

This thesis is organized in six parts. Each part is divided in chapters, in which We try to address some issues that often arise in the previous ones. Each chapter is meant to be self-explanatory, thus providing an introduction to the problem, related works, system development and conclusions.

This first part provides motivations, background information and a literature review of the study relative to the employment of attentional systems in robotics. In this section I also try to underly the contributions of this thesis with respect to the systems present in literature.

In the second part, I show the designed attentional system for a robotic agent capable of adapting its emergent behavior to the surrounding environment and to

its internal state, by optimizing resource utilization. In this framework, the agent is endowed with simple selective attentional mechanisms regulating the percentage of attention towards internal or external stimuli. The realized system is also endowed with some kind of divided attention mechanism able to opportunely split resources among concurrent behavior. The framework is presented by discussing several case studies, considering incrementally complex behaviors and tasks, in which I try to emphasize the benefits introduced by this kind of technics with respect to behavior-based standard control systems.

The third part concerns with the adoption of some evolutionary approach, used to suitably tune the key parameters regulating the attentional system. In this part I also present the framework, implementing it by means of a neural network, on which I apply the same learning technique.

Part four deals with the design of a hybrid architecture in which attentional mechanisms are deployed at different levels of the architecture. Here I want to show how the adaptability introduced by the attentional system allow to deal with dynamic environment, characterized by unpredictable events.

In part five I show an application presented in order to investigate the use of our approach, and verify its utility in experiments involving human-robot interaction tasks.

Finally, the last part contains concluding remarks and proposals for further investigations. Here it is stated that the executive control system is effectively enhanced through the use of a supervisory attentional system, able to suitably combine deliberative and reactive activities, monitoring and regulating multiple concurrent behaviors.

Part II

Reactive Attentional System

Chapter 2

Selective Attention Mechanism

2.1 Introduction to Selective Attention

The Selective Attention represents the process by which organisms select a subset of available information upon which to focus. This ability to select a small fraction of the incoming sensory information enhances performance [56], permitting to reduce the computational load in analyzing environmental scenes and in planning responses coherently with behavioral goals. Inspired by the attentional abilities of human beings, we seek to benefit from the use of these mechanisms for improving a robotic control system. We apply such type of mechanism within a robotic control system in order to obtain the advantages of both filtering the available information and improving performance.

2.2 Motivations

An *intelligent* connection between perception and action needs mechanisms for controlling action execution, in respect of the constraints imposed by the mechanical system and the environment and for providing some feedback process, which regulates the internal states of the system arising from the interaction with the environment. Such a type of connection system has to combine different low-level strategies with high-level activities, giving them, from time to time, different priority values both for resource allocation and for action selection processes. The

low-level activities are closely related to the safety of the system, and may be achieved by applying principles of reactive control. However, high-level activities generally are achieved by processing more complex tasks, and, thus then require high computational costs for both the inputs processing and the acquisition of data from the environment. Also with hybrid systems, in which perceptual information can modify the planning of actions, sensor readings and planning activities must occur with a frequency that does not excessively slow down the Robotic System (RS), since increasing the number of readings increases the number of accesses to the deliberative system. At the same time, however, we must keep in mind that the RS takes risks if one allows sensor readings with long time intervals. In fact, during the planning activity, in between two consecutive sensors readings, the environment may be changed and, therefore, the RS may no longer behaves properly. In this sense, any effort to build a cognitive architecture for dealing with dynamical and flexible emergent behaviors has to deal with an efficient processing capability of sensor elaboration. The robotic community started to pay attention not only to the robot-environment interaction, but also to the interaction that may arise within the robots itself [57] and how these latter (for example its emotional state) may have some influence on its emergent behavior. The attention for such *internal mechanisms*, within the robotic community, takes inspiration from ethological, biological and neuroscience studies.

In our opinion the internal mechanisms which have to be involved in modeling different and new architectures for controlling the robot behavior, and try to solve some of the problems inherent with the management of computational resources are the *attention mechanisms*. In particular, the bottom-up attentional processes could be used to manage the interaction occurring between the robot and the surrounding environment, while the top-down ones could be usefully adopted to manage the interaction arising from its internal states. We also think that these processes have to be linked since, for example, the simple perception-action response to an external stimulus may produce different patterns of actions as a consequence to a different internal state of the robot. This internal state may change according to the robot emotional state or following its past perception and it will tune and adapt both the behaviors execution and the sensory processing frequency.

We will introduce our approach to model adaptive control systems [58], [59],

based on the concept of rhythmic activations of behaviors, with the aim to experiment how it is possible to develop adaptive strategies that involve the presence of attentional internal mechanisms.

The idea to use *rhythmic activation* in order to coordinate sensory-motor activities comes from studies on our nervous system, whose main function is the coordination or integration of the activities of the various parts of the body [60]. Concerning this topic, there are some suggestions regarding the neural integration, which is supposed to have a central role in the origin of respiratory rhythm in fish [61]. In 1935 Weiss proposed that the central nervous system produces rhythmic motor commands, with no need of sensory feedback; the main evidence came from larvae of amphibians [62]. Then, in 1939, the German physiologist von Holst discovered the existence of endogenous neural oscillators that coordinate the rhythmic activities of organic systems and the two principles governing this coordination: (1) the absolute coordination, i.e. the tendency of an oscillator to maintain a steady pace leading to fully synchronized movements ("absolute coordination" states); and (2) the "relative coordination", i.e. the effect an oscillator exerts on the frequency of another oscillator in a way that it seems it magnetically attracts and "relates" the other to its frequency [63]. We will use both these concepts to create our control system made by independent self-regulating behaviors and some mutual-influence rules to synchronize behavior rhythms in case of concurrent tasks. In addition, we bind the concept of synchronization to the concept of attention. In fact, synchronization of neural firing may also be the basis of the process of attention. While it is attentional, the brain selects certain stimuli or events that provide attention to be preferred. Research suggests that this selection synchronization can occur through strengthening the neural firing in some groups of neurons, while it decreases in others [64], [65]. Hence we adopt the concept of attention for regulating sensory-motor coordination, implementing it by starting from biological evidences, such as rhythmic activation observed in neurological studies on human beings.

Our approach, resulting from the need of an Internal Robotics [57], should allow mediating between the ideas of the Situated Cognition [66] and the need of using simple representation constructs, in order to link adaptive behavior with high level cognition processes.

2.3 Model of Selective Attention

Our working hypothesis, supported by neurological and behavioral studies, is that attentional behaviors are affected by internal self-regulating mechanisms and external sources of salience, and the attentional global behavior emerges from the interrelation of the attentional mechanisms associated with each single behavior.

More precisely, we think that attentional behaviors can be simulated in the control activity of a robot starting from self-regulating mechanisms. This goal can be achieved by introducing *internal rhythmic clocks* in robotic architecture. Such clocks are meant to regulate the frequency of the readings in a way that the process of regulating the frequency of sensory readings can be interpreted as an increase or decrease of attention towards salient sources of information depending on the behavior.

Also neurological studies show that the simplest selection process is a rate-based mechanism, in which the responses of neurons in early processing stages that convey information to be selected are made more prominent by raising their firing rates, whereas the responses of neurons that convey information to be ignored are made less prominent by suppressing or decreasing their firing rates [64]. Hence, what we want to show is that the selection of sensory information obtained by modifying behaviors' activation rates is a powerful way of affecting the relative importance of different sources of information [67].

We introduce a control system for the perception inputs that achieves a rhythmic and flexible activity and thus it dynamically adapts its period to external and internal requirements. In particular, we connected a periodic control system to the activation of each single behavior [68]. Our aim is to investigate how the introduction of such an implementation of an attentional mechanism into the controlling system of each single behavior will affect the elaboration of perceptual data. In particular, the question we have to answer is twofold. First, is the robot we are building safe? This means that the robot is able to react, in useful time, to the external stimuli in order to survive also when we break the stimuli response loop and modify the frequency of access to the sensor inputs. Second, will the introduction of attentional filters within the behavior control system provide better performance, in terms of a reduction of computational load?

We make the assumption of a cognitive architecture with a perceptual system and with some releasing mechanism of activation of behaviors.

2.3.1 Adaptive Innate Releasing Mechanism: AIRM

Our architecture combines innate releasing or inhibiting mechanisms and simulated biological clocks in order to produce attentional mechanisms.

Innate releasing or inhibiting mechanisms: Lorentz [69] and Tinbergen [70] identified in many animals an innate releasing mechanism (called IRM) able to control and coordinate behaviors. An IRM is based on a specific stimulus that releases a pattern of actions. For example, an animal may have a pray as an IRM, i.e. the stimulus coming from the view of the predator which activates the escape behavior. IRMs were included in the representation schema of behaviors in the form of releasers, controlling when behaviors must be activated or deactivated. A releaser is an activation mechanism that depends on exogenous factors (e.g. presence of a predator) and/or endogenous factors (e.g. hunger).

Simulated biological clocks: The releasers function, somehow, recalls the notion of "internal clock", already introduced in some approaches [53], [52], [71] in order to activate motivational states for a robot (for example, hunger or sleep). In fact, an internal clock, similarly to a releaser, represents an internal mechanism which regulates behaviors activations [72] depending on endogenous and/or exogenous factors.

For these similarities we called these simulated biological clock: *Adaptive Innate Releasing Mechanisms* (AIRMs). However, there are substantial differences between IRMs and AIRMs; one over all is that while a releaser is an instantaneous activation mechanism, the internal clock is periodical and adaptive.

Indeed, an internal clock may imply a regular and periodical activations of the associated behavior. Such activations may be predicted in time, while the activity of a releaser depends only on contingent factors. In this way no computational resources are spent to elaborate unneeded stimuli, because the corresponding control systems are kept *inactive* until a new periodical activation takes place. At the same time we are able to control the amount of resources spent in the elaboration of the sensor inputs.

Moreover, the introduction of internal clocks within a robotic architecture has also the effect of controlling behaviors that may require a fixed pattern of activation in time (like sleeping or feeding). This activation of behavior may be interpreted both as large time scale activities, such as the activations of macro-behaviors like feeding or sleeping, and as short time scale activities, in the sense of central-pattern generators in controlling rhythmic movements.

2.3.2 Formalization of the AIRM model

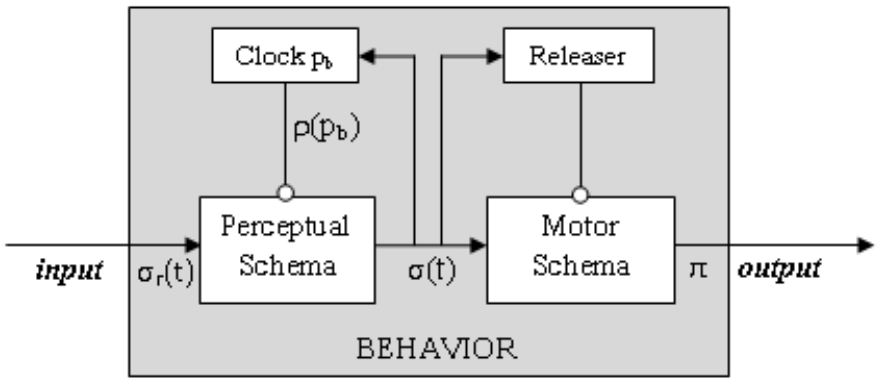


Figure 2.1: Behavior endowed with Adaptive Innate Releasing Mechanism.

In Figure 2.1 the AIRM is represented through a Schema Theory representation [73]. Each behavior is characterized by a schema composed of a Perceptual Schema (PS), which elaborates sensor data, a Motor Schema (MS), producing the pattern of motor actions, and a control mechanism, based on a combination of a clock and a releaser.

In particular, the releaser enables/disables the activation of the MS, according to the sensor data $\sigma(t)$. For example, the presence of a predator releases the motor schema of an escape behavior. In this way the MS is activated only in the presence of the stimulus, while sensing data are always (i.e. at each machine cycle) processed from PS.

In contrast the AIRM directly enables/disables data flow $\sigma_r(t)$ from sensors to PS and thus, when the activation is disabled, sensing data are not processed

(yielding to sensory readings reduction). Furthermore, the internal clock regulates the frequency of the activations, hence the frequency of data processing (behavior adaptation),using a feedback mechanism on the processed sensing data $\sigma(t)$.

We assume a discrete time model — with the machine cycle as the time unit — where each behavior is endowed with a clock regulating its own activations. This regulation mechanism, that we call *monitoring strategy*, is characterized by:

- A period p_b, where b is the behavior identifier (e.g. its name), that is initially set equal to a given starting period pi_b called base period, ranging in an interval $[p_bmin,\ p_bmax]$,

- An *updating function* $f_{a,d}(\sigma(t), p_b^{t-1}) : \mathbb{R}^n \rightarrow \mathbb{R}$ that adjusts the current clock period p_b^t, according to the internal state of the behavior and to the environmental changes. In particular we distinguish the case of an increasing $(f_a(\sigma(t), p_b^{t-1}))$ and reducing $(f_d(\sigma(t), p_b^{t-1}))$ updating function, where $\sigma(t)$ is the incoming signal from sensor and p_b^{t-1} is the value of the period computed at the previous sampling time.

- A trigger function $\rho(t, p_b^t)$, which enables/disables the data flow $\sigma_r(t)$ from sensors to PS at each p^t time unit. More formally:

$$\rho(t, p^t) = \begin{cases} 1, & \text{if } t \bmod p^t = 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

- Finally, a support function $\phi(f_{a,d}(\sigma(t), p_b^{t-1})) : \mathbb{R} \rightarrow \mathbb{N}$ that maps the values generated by the updating function $f_{a,d}(x)$ in a range of allowed values for the period $[p_bmin,\ p_bmax]$. More precisely:

$$\phi(x) = \begin{cases} p_{max}, & \text{if } x \geq p_{max} \\ \lfloor x \rfloor, & \text{if } p_{min} < x < p_{max} \\ p_{min}, & \text{if } x \leq p_{min} \end{cases} \tag{2.2}$$

Now, starting from the clock period at time 0, $p_b^0 = pi_b$ (with $t = 0$ and $pi_b \in [p_bmin,\ p_bmax]$), the clock period at time t is regulated as follows:

$$p_b^t = \rho(t, p_b^{t-1}) * \phi(f_{a,d}(\sigma(t), p_b^{t-1}) + (1 - \rho(t, p_b^{t-1})) * p_b^{t-1} \tag{2.3}$$

That is, if the behavior is disabled, the value of the period calculated at time t remains unchanged at the last computed value p_b^{t-1}. Instead, when the value of the trigger function is equal to 1, the behavior is activated and, subsequently, its activation period changes according to the $\phi(x)$ function.
More precisely, each time the behavior is activated and hence has enabled the data flow, the sensory information is passed through a feedback mechanism to the internal clock control system, which updates the clock period depending on internal and environmental conditions. The mechanism for period/rate regulation is called the *monitoring strategy* and will be detailed in the following section. The monitoring strategy, i.e. the process of changing the clock sampling rate, can be associated with the increase or decrease of attention towards a particular behavior. Namely, the more salient the behavior, the higher the clock frequency and the resolution at which a behavior is monitored and regulated. Intuitively, the mechanism to focus the attention towards a particular stimulus (i.e. reduction of the period) can be different from the mechanism of distraction (i.e. increment of the period). This is why we choose to distinguish the two cases through different updating functions, at which we will refer respectively as *focusing* $f_a(x)$ and *distraction updating function* $f_d(x)$.

2.3.3 Attentive Monitoring Strategy

From the above description it follows that an attentive behavior will result from the combination of:

- the initial period p_i;
- the range of allowed values for the period $[p_b min, p_b max]$;
- the updating policies respectively for attentional $f_a(\sigma(t), p_b^{t-1})$ and distraction $f_d(\sigma(t), p_b^{t-1})$ phases.

The combination of these parameters defines what we call *monitoring strategy* and thus the policy for scheduling sensing activities. In order to obtain a good mon-

itoring strategy, it is necessary to balance the cost of monitoring and the risk of inaccurate and partial information about the environment, by choosing the appropriate updating function. For example, the two main updating functions we use for the attentive phase take respectively into account the speed with which environmental changes occur $f_{a_1}(\sigma(t), p_b^{t-1}) \backsimeq \frac{\Delta\sigma(t)}{p_b^{t-1}}$, and how much the sensorial stimulus is changed with respect to the previous quantity perceived $f_{a_2}(\sigma(t), p^{t-1}) \backsimeq \frac{\Delta\sigma(t)}{\sigma t}$, where $\Delta\sigma(t) = \sigma(t) - \sigma(t - p_b^{t-1})$, $\sigma(t)$ is the signal perceived at the current time and $\sigma(t - p_b^{t-1})$ is the sensor signal received at the previous sampling instant. Yet, we can choose many different function for the update. Following this approach, we can obtain different attentional mechanisms associated with each behavior, once we define the associated monitoring strategy.

Just to give an idea of the general functionality of this mechanism, we compare our monitoring strategy (adaptive and periodical) with respect to other relevant monitoring strategies proposed in the literature. In particular, we consider four different cases: a) the robot is not equipped with any internal clock; b) the robot is equipped with internal clocks and has a priori knowledge about the task to achieve (for example about the distance to cover); c) the robot is equipped with clocks, but does not have any a priori knowledge about the environment; d) the robot is equipped with internal clocks but not adaptive.

To illustrate these cases we introduce the following example. Let us consider a robotic system whose purpose is to cover a certain distance (*goTo* behavior).

Without an internal clock (a), *goTo* will be activated at each control cycle. If the covered distance is constant at each control cycle, we have that the number of activations n of *goTo* is proportional to the distance *dist* to be covered (see Fig. 2.2). In case (d), if the value of the clock period is p_b, we have the relation: $n \propto \frac{dist}{p_b}$. Cases (b) and (c) fall in the category of those strategies called "Interval Reduction" [74], which are characterized by a variable period for the monitoring strategy. For cases (b) and (c), it has been demonstrated that these strategies, asymptotically, are more effective than those characterized by a constant periodic monitoring [74, 75] in a wide class of problems. Moreover, in this setting, an interval reduction strategy has to increase the behavior activation frequency while approaching the goal. If the robot is equipped with adaptive clocks and knows a priori the distance to cover, we might set the initial period $p_b = pi_b$ proportional

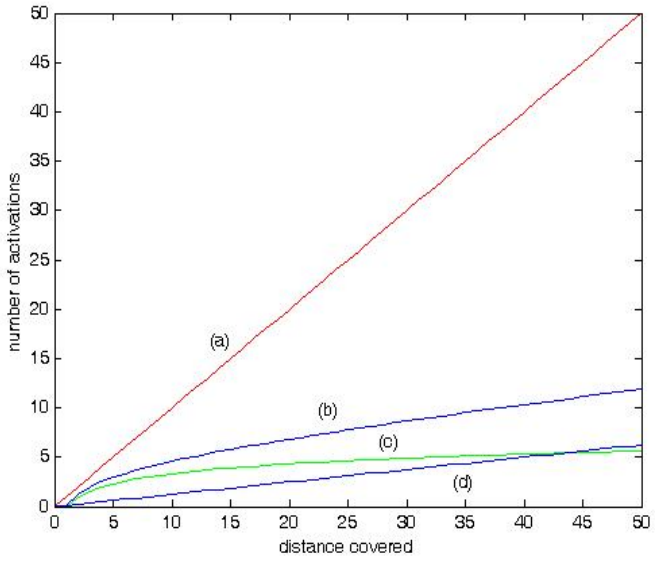


Figure 2.2: Monitoring strategies for *goTo* behavior. Number of activations in the case of: (a) continuous monitoring; (b) periodic and adaptive monitoring without a priori knowledge; (c) periodic and adaptive monitoring with a priori knowledge; (d) periodic monitoring.

to half of the distance to cover and then halve the period of the clock after each activation of the behavior following the interval reduction strategy. In this way, the number of activations would be: $n \propto log_2(dist)$. We assume that this strategy is the "optimum", where for optimum we mean a strategy that allows achieving the goal through the minimum number of activations, without the risk of jeopardizing the correct and efficient functioning of the robotic system and its safeguard. Let us now describe how the robot behaves in case (c), namely, when it is characterized by adaptive rhythms, but without a priori knowledge. In this case, the rhythm must change gradually in accordance with a law that does not diverge too far from the optimum case. In fact, the robot, even if not provided with a priori knowledge, can obtain information from the surrounding environment, thanks to the use of its sensors and, through these values, it may determine the choice of the rhythm. For example the number of activation in this case may be approximated as: $n \propto log_2(dist) + (dist_{cov})/p_b$, where $dist_{cov}$ is the distance already covered by the robot. We can see that the number of activations in case (b) will have as an

upper bound case (a), and as a lower bound case (c).

Overall, the benefits brought by adaptive and periodical monitoring strategies are mainly two:

- *periodical mechanisms* of activation can reduce the number of activations of the behavior (with respect to the standard case (a) in which the activations are performed at every machine cycle), causing a relative decrease in the computational burden, and improving performance of the entire system;

- the use of *adaptive mechanisms* allows us to obtain a behavior that adapts itself to the specific environmental conditions (e.g. the robot reads sensors more often if there is a dangerous situation or it needs more precise information and less often in case of a safe operational situation or distraction).

Afterwards, we will show that, by calibrating appropriately the basic rhythms and using the appropriate policies to update them, we can obtain a significant improvement in performance compared to an architecture without rhythms.

2.3.4 Design principles overview

In summary, the attentional control system we consider in this work combines the following design principles:

1. *Behavior-based control system.* The attentional control is obtained from the interaction of a set of multiple parallel attentional behaviors working at different levels of abstraction.

2. *Attentional monitoring.* Attentional mechanisms are able to focus monitoring and control activities on relevant internal behaviors and external stimuli.

3. *Internal and external sources of salience.* The sources of salience are generally behavior- and task-dependent; these can depend on either internal states (top-down data stream) e.g. hunger, fear, reaching a goal position, etc. or external stimuli (bottom-up data stream) e.g. obstacles, unexpected variations of the environment, attractiveness of a particular object, etc..

4. *Selective Attention: adaptive sensory readings.* For each behavior, the process of changing the rate of sensory readings is interpreted as an increase or decrease of selective attention towards a particular aspect of the environment the robotic system is interacting with: the higher the frequency, the higher the resolution at which an activity is monitored and regulated.

5. *Divided Attention: mutual influence rules.* The adaptive frequency of the sensory sampling rates provides, by means of specific mutual influence rules, a kind of divided attention: the activation of a behavior can lead directly to an increase or decrease in the rate of activation of behaviors related to it, producing an homeostatic effect, according to which shared resources are appropriately distributed among conflicting behaviors.

6. *Emergent attentional behavior.* The overall attentional behavior should emerge from the interrelation of the attentional mechanisms associated with the different primitive behaviors.

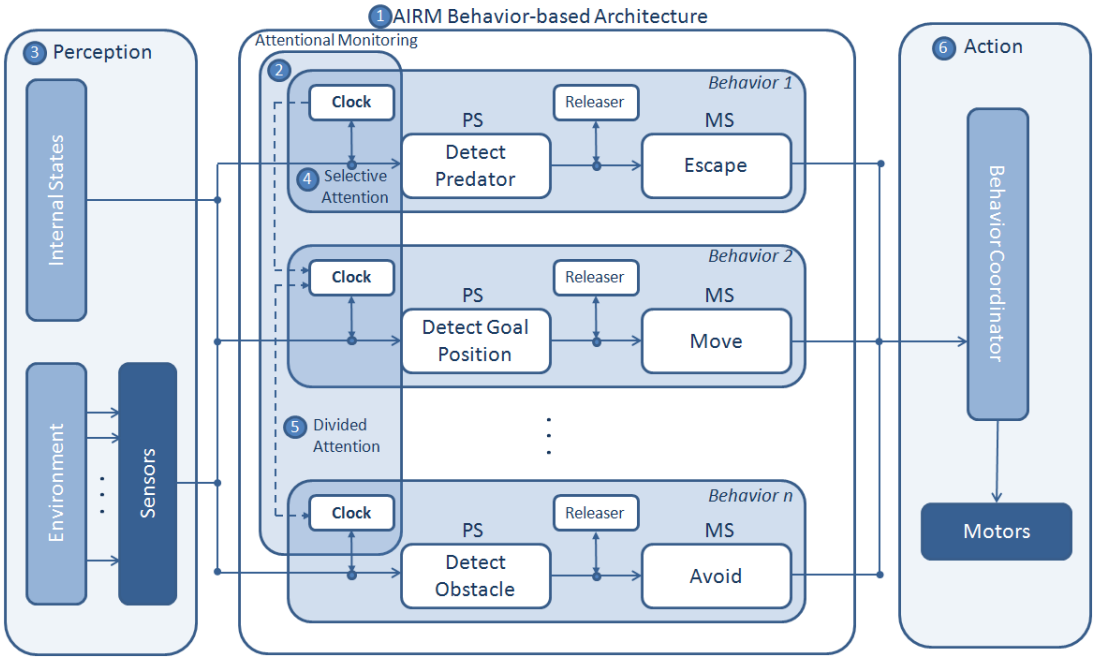


Figure 2.3: AIRM Architecture Overview.

2.4 Test-bed Case Study 1

Based on the model introduced in the previous section, we designed a behavior-based robotic system which use different attentional monitoring strategies in order to adaptively regulate the attention towards the different behaviors.

We used a PIONEER 3DX, an indoor research platform commercialized by MobileRobots [76]. It is an electric-drive robotic system with multiple on-board sensor systems: laser, pan-tilt camera, ultrasound sensors,etc. In our experiment we use a set of sensors composed of: a blob camera, an odometer and 16 sonar sensors.

Figure 2.4: Pioneer3DX.

The base Pioneer 3DX platform arrives fully assembled with motors with 500-tick encoders, 19cm wheels, tough aluminum body, 8 forward-facing ultrasonic (sonar) sensors, 8 optional real-facing sonar, 1, 2 or 3 hot-swappable batteries, and our complete software development kit. Add an optional internal computer or your own laptop and the robot is ready to go. The base Pioneer 3DX platform can reach speeds of 1.6 meters per second and carry a payload of up to 23 kg. Pioneer is fully programmable. In particular we control it by means of the Player/Stage robotic tool [77].

2.4.1 Cataglyphis Ant Domain

We evaluated our approach using a mobile robot that simulates the navigation behavior of a Cataglyphis ant enhanced with simple visual capabilities. The robot has the task of searching food in its environment without any *a priori knowledge* and then return to its nest following a straight path, without taking into account the trajectory followed during the searching of food. The Cataglyphis domain is a good test-bed for our purposes since the domain is interesting from a behavioral point of view and well analyzed in several ethological field trials [78]; furthermore the ant is provided with internal mechanisms such as guidance and dead-reckoning. So, we can associate with this test-bed all the monitoring strategies previously presented (constant, with a priori knowledge, without a priori knowledge). At this level the domain chosen, is not important since it is just an example of multi-behavior system in which we can associate different monitoring strategies to each behavior. The behavior-based architecture realized is shown in Fig. 2.5. It is characterized by three meta-behaviors which described the main phases of the task. Each behavior is of the type described before. That is, it receives data from some of the sensors and generates the consequent action, only when the releaser and simultaneously the clock decide the behavior may be activated. Hence, besides the implementation of the perceptual and motor schema functions, we have to define the monitoring strategy of each behavior.

2.4.2 Attentive Architecture Overview

The robot behavior is obtained as the combination of the following primitive behaviors `AVOID`, `WANDER`, `PATH_INTEGRATION`, `MOVE_TO_FOOD`, `MOVE_TO_NEST` and `FIND_LANDMARKS`, organized in three meta-behaviors (Fig. 2.5). More precisely, the behaviors are combined through the classic mechanisms of the subsumption architectures Brooks-like [7], but here the emergent behavior is also affected by the rhythms of the behaviors activation.

Behaviors settings and Attentive Updating policies. For each behavior, we have to define the *attentional monitoring strategy* composed of: 1) an *updating policy* and the 2) a *base period*, that is empirically defined after a phase of testing.

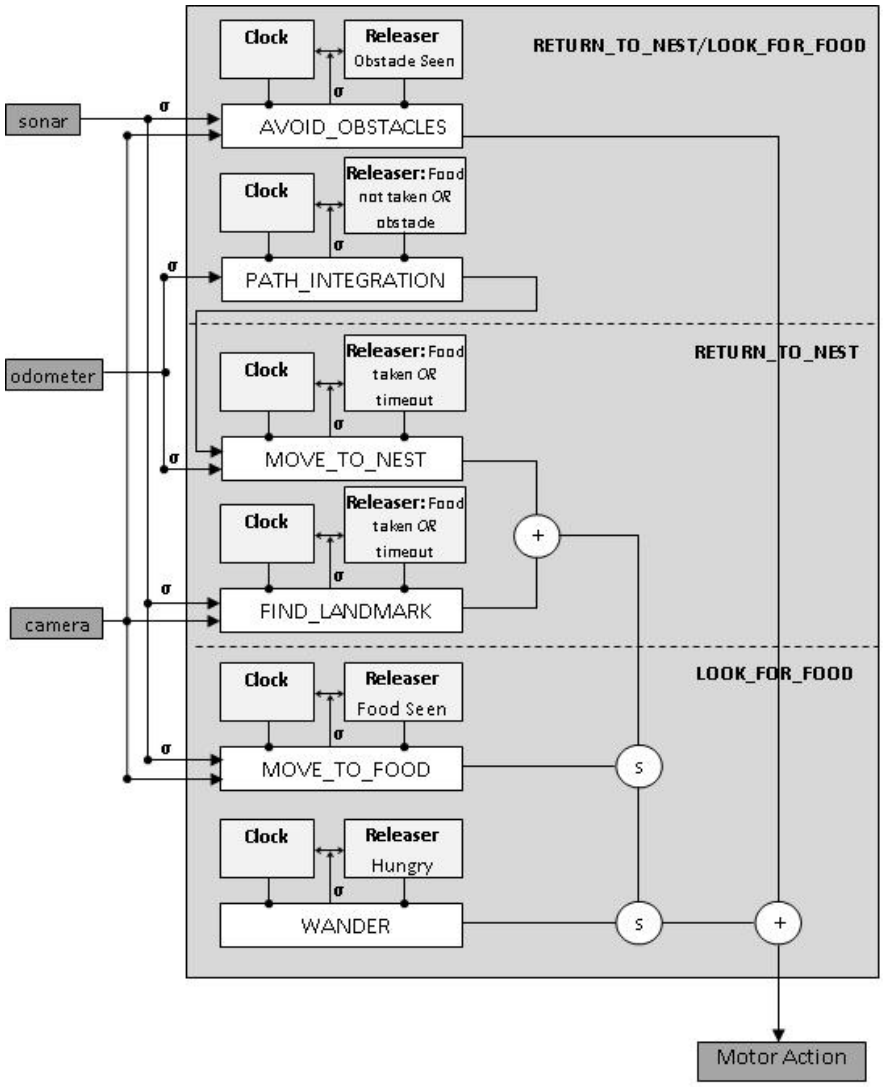


Figure 2.5: Control architecture for the mobile robot in the Cataglyphis domain. Each behavior receives data from sensors and generates the actions that can be combined (circled +) or subsumed (circled s).

In the third part of this thesis we will see how it is possible to learn both these parameters and the updating policies by means of some specific learning algorithm.

Let us start describing each behavior and the associated *updating policy*. The `WANDER` behavior provides a random search in the environment. Since its activation is periodic, but not adaptive, this can be associated with a constant clock.

Furthermore, this behavior is not critical and always in background, therefore we can slow down the clock frequency minimizing the behavior activations

$$p_w^t = const$$

The output of this behavior is a random pattern of orientations for the motor action.

In contrast, the `AVOID` behavior, responsible for obstacle avoidance, is safety critical and needs an adaptive clock and an associated updating policy to timely react to dangerous situations.

We can imagine that the attention towards the obstacles starts when the agent detects a particular obstacle and continuous to increase proportionally to the proximity of the interested object until the obstacle is not avoided. While the distraction starts when the agent has not to pay attention since there are not other obstacle in front. In this case, while the clock period increment can occur in a linear fashion, the decrease must be proportional to the seriousness of the situation of danger. So we update the AVOID period according to the first derivative of the sensory input (representing the distance from the nearest object, evaluated by the sonar sensors), with respect to the time occurred between two consecutive sampling readings. Intuitively, the clock frequency is adaptive with respect to the speed at which the environmental changes occurs in a way that the higher the change, the smaller the sensor sampling rate. This is useful since in a dynamic environment the robot might suddenly find itself in front of an unexpected obstacle, and in this case it would be more appropriate to change the rhythm of reading in proportion not only to the change, but also in proportion to the speed at which this happened. Of course this is a possibility, but we can choose for each behavior the more appropriate strategy (linear, logarithmical, exponential and so on).

More formally, the policy is to change the `AVOID` clock period according to the first derivative of the sensory input $\sigma(t)$, that represents the distance from the nearest object, evaluated by the sonar sensors. The period p_a^t is updated with the following focusing function:

$$f_a(\sigma(t), p_a^{t-1}) = \frac{\sigma(t) - \sigma(t - p_a^{t-1})}{p_a^{t-1}}$$

and according to (2.3):

$$p_a^t = \rho(t, p_a^{t-1}) * \phi(f_a(\sigma(t), p_a^{t-1}) + (1 - \rho(t, p_a^{t-1})) * p_a^{t-1}$$

where p_a^{t-1} is the period at the previous behavior activation, $\rho(t, p_a^{t-1})$ is the releasing function that enables the sensory sampling, φ is a normalizing function mapping the derivative into a set of permitted values, and $\sigma(t - p_a^{t-1})$ is the value of sensor at the previous behavior activation. Intuitively, the clock frequency is adaptive with respect to the environmental changes: the higher the change, the smaller the sensory sampling. In this way, the activation frequency adapts itself not only to the environmental changes, but also to the speed at which these changes take place.

The distraction function can be expressed through a linear function

$$f_d(p_a^{t-1}) = p_a^{t-1} + const_a$$

The `AVOID` behavior is responsible not only for the robot orientation, but also for its speed variations. In particular, speed is related to the period according to the relation

$$speed = \frac{max\_speed \times p_a^t}{p_a max},$$

where *speed* is the current speed, *max_speed* is the maximum value allowed for the robot speed. The range of values for the speed is $[0, 0.3]$ m/s. In this way, if the period is relaxed, the robot moves at a maximum speed, otherwise slows in proportion to the decrease of the period. This allows the agent to avoid obstacles in a smooth way (see the next sections for details).

In Fig. 2.6, we can see how the avoidance period changes over time, based on the variation in sensors readings. The red line represents the sonar value, the blue line the change in velocity, and the green bars represent the activations of the avoid behavior. The blank space between two consecutive green bars is the actual period of the internal clock of the avoidance behavior. What we see is that to a substantial change in sonar readings corresponds a proportionally reduction in the period value. And what we want to show is how, appropriately setting the basic periods and using the appropriate policies to update them, we can coordinate

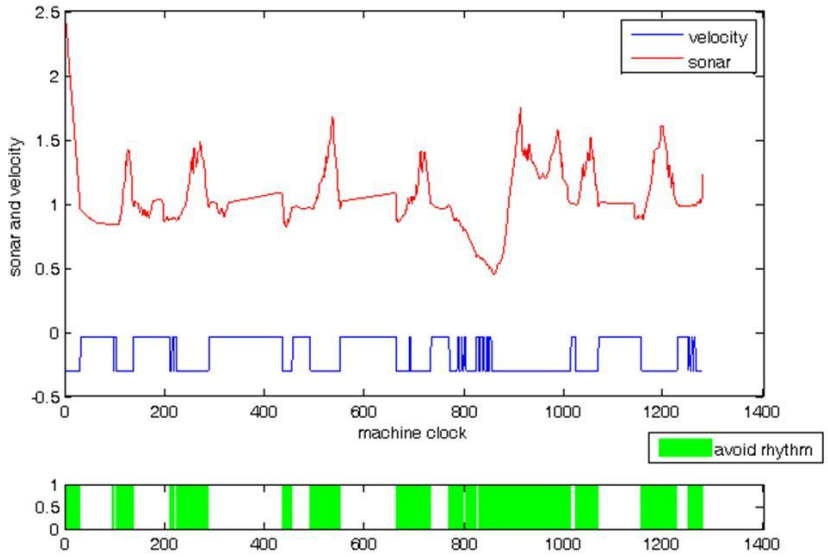


Figure 2.6: Changes of the avoidance period with respect to the variation in sensors reading.

sensor and action and we can obtain adaptive attentional behavior, by achieving also a significant improvement in performance.

`MOVE_TO_FOOD` detects the food and guides the robotic system towards it, setting its direction. If the releaser is on and the agent sees the food, then the output will be a movement towards the food, otherwise the agent will produce a random movement aimed at finding food. In order to obtain reliable information from the camera, this behavior involves the robot to slow down its velocity. Thus, when it is activated, it reduces speed system. Choosing an adaptive clock period for this behavior, i.e. reducing the number of behavior activations, we allow the robot to reach the food as soon as possible, but we have to set the base period and the updating policy taking into account that we want to achieve the goal with the minimum docking error. Thus we ought to balance effectiveness and precision. The idea is to keep the base period pi_m constantly equal to its allowed *maximum value* $p_m max$ until the robot reaches a fixed *distance threshold*, so that the activations are minimized until the robot does not achieve that threshold, and then decrease the

period linearly with the food distance according to the following focusing function:

$$f_a(\sigma(t), p_m^{t-1}) = \sigma(t) - \sigma(t - p_m^{t-1})$$

where $\sigma(t) - \sigma(t - p_m^{t-1})$ represents the distance at time t. Here, the behavior performance will depend on the optimal balance between the clock frequency and distance.

`PATH_INTEGRATION` manages the direction and the distance to return to nest. This behavior uses odometric sensor information to calculate the robot shift with respect to the previous position. Analogously to `WANDER`, the behavior remains always active with a constant clock period, until it reaches food. In the return phase, the behavior is activated only after an abrupt change of course, e.g. due to the presence of obstacles, which diverts the trajectory of the robot that has to be recalculated. Therefore, we can state that its focusing function is generally equal to

$$f_a(\sigma(t), p_p^{t-1}) = p_p^{t-1}$$

`RETURN_TO_NEST` sets the direction of the robot toward the nest. It requires, as the `MOVE_TO_FOOD` behavior, reliable information from the camera. It slows down the robot velocity while its perceptive system is active. This behavior exploits a priori knowledge (i.e. the distance to the nest) that allows us to set the base period value proportional to half the distance from the nest, $pi_r = distanceToNest(t_0/(2k))$ and then reduces accordingly.

It has been shown that, asymptotically, this strategy is very efficient [74], [75] (see section 2.3.3).

Wherever they are not specified, the distraction updating functions decrease the period linearly with time.

Setting the base period. In this first implementation of the architecture we defined the attentive parameters experimentally. Below we show an example of how you can experimentally select the appropriate value of the base period, for example for the `MOVE_TO_FOOD` behavior. In part 3 of this thesis, we will show suitable learning mechanism to automatically set the attentive parameters regulating the

monitoring strategy adaptively with respect to the environmental dynamics. To set the maximum value $p_m max$ for the base period pi_m associated with `MOVE_TO_FOOD`, we tested the behavior performance under different salient conditions; the behavior performance depends on the optimal balance between the clock frequency and threshold distance from food, selected in order to start reducing the period. As mentioned previously, the updating policy is to keep constant the *clock period* until a certain *distance threshold*, then linearly reduce it. The parameters we have to establish a priori are the *maximum base period* and the *distance threshold*. These are chosen on the basis of both the environmental and robot features, trying to optimize the performance and simultaneously taking care of the security constraints. Once we have selected these two parameters, we test the `MOVE_TO_FOOD` behavior performance under different salient conditions, with the aim of establishing what is the suitable base period pi_m in each situation; the behavior performance depends on the optimal balance between the clock frequency and threshold distance from the food, selected in order to start reducing the period. We assume robot and food far from obstacles. We test the behavior with three different initial values of the distance from food, and compare the result obtained with all possible allowed value for the base period [1,*maximum base period*]. In Fig. 2.7, we show the values for the docking error and the total amount of time for the behavior activations, obtained with some possible combinations of $p_m max$ and threshold distances (assuming robot and food far from obstacles). We report the average and variance of the values gathered in 10 runs for each case. For each configuration, the initial distance from the food is equal to the threshold distance. In addition a horizontal line separates safe/unsafe settings: below the line there are settings where the robot can stop beyond a safety distance from the target, hence incurring in dangerous situation.

Looking at the results, we notice for example that, if the initial distance from food is equal to 120 cm the *base period* which yields the best performance is $p_b = 8$, because it allows minimizing the time of behavior activation with a very small docking error, which is not excessive from being in dangerous situations (horizontal line separates safe/unsafe settings: below the line there are settings where the robot can stop beyond a safety distance from the target, hence incurring in dangerous situation). Similar performance is obtained also for the combination

	distance = 200 cm	
$p_m max$	activation time (s)	docking error (cm)
1	20.274 ∓ 0.630	0.178 ∓ 0.018
2	7.638 ∓ 0.480	$0.28 \mp 2.37e^{-004}$
4	5.116 ∓ 0.884	$0.28 \mp 1.17e^{-004}$
8	2.991 ∓ 0.311	$0.28 \mp 2.17e^{-004}$
16	1.213 ∓ 0.452	0.250 ∓ 0.004
32	1.292 ∓ 0.363	0.085 ∓ 0.0019
	distance = 120 cm	
$p_m max$	activation time (s)	docking error (cm)
1	7.759 ∓ 1.320	$0.289 \mp 9,25e^{-01}$
2	3.667 ∓ 0.274	$0.287 \mp 3,11e^{001}$
4	3.046 ∓ 0.145	$0.292 \mp 3,25e^{-01}$
8	1.392 ∓ 0.335	$0.231 \mp 4,25e^{-01}$
16	0.649 ∓ 0.059	0.195 ∓ 0.002
32	40.45 ∓ 0.053	$0.123 \mp 5,7e^{+001}$
	distance = 60 cm	
$p_m max$	activation time (s)	docking error (cm)
1	0.149 ∓ 0.006	$0.215 \mp 1.125e^{-004}$
2	0.194 ∓ 0.003	$0.206 \mp 6.75e^{-005}$
4	0.167 ∓ 0.004	$0.173 \mp 1.57e^{-004}$
8	0.237 ∓ 0.004	$0.116 \mp 6.30e^{-004}$
16	0.167 ∓ 0.004	$0.020 \mp 2.83e^{-004}$
32	0.199 ∓ 0.003	$0.012 \mp 2.70e^{-006}$

Figure 2.7: An evaluation of the `MOVE_TO_FOOD` behavior, in terms of activation time and docking error, varying the distance and the max value for the base period pi_m.

distance = 200 cm and $p_b = 16$. Looking at the docking error and the total amount of time for the behavior activations obtained in each test, we can choose the optimum base period by balancing the tradeoff between safety and efficiency. We will extend the architecture in order to automatically learn these values.

System assessment. To assess the system performance, we compared the adaptive control system with respect to a non-adaptive one (i.e. with sensor readings fixed at every machine cycle). Our aim is to show that a significant improvement in performance can be obtained by appropriately tuning the basic periods of clocks

and the attentional policies to update them. Therefore, for each specific behavior we will evaluate the updating policies, by seeking, on one hand, to optimize performance (i.e. less sensor readings) and, on the other hand, to enhance robot safety and the correctness of the overall system behavior.

2.4.3 Experimental Results

In order to assess the system performance, we compared the system behavior when endowed with adaptive clocks with respect to a not adaptive periodic version (e.g. activations at the machine clock). In particular, for each behavior, we considered the number of activations and the total amount of time spent in behavior execution.

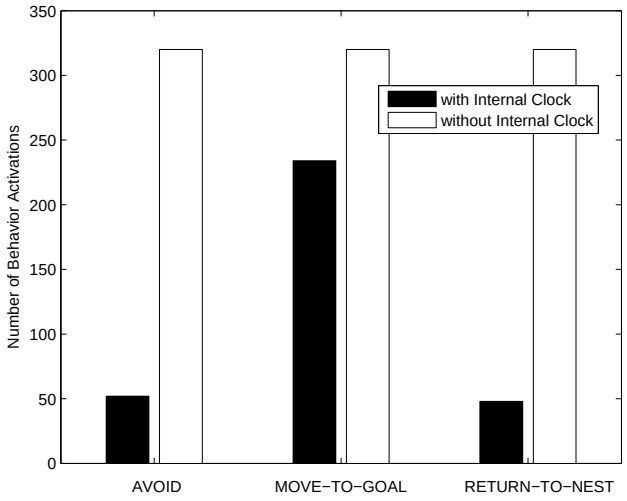
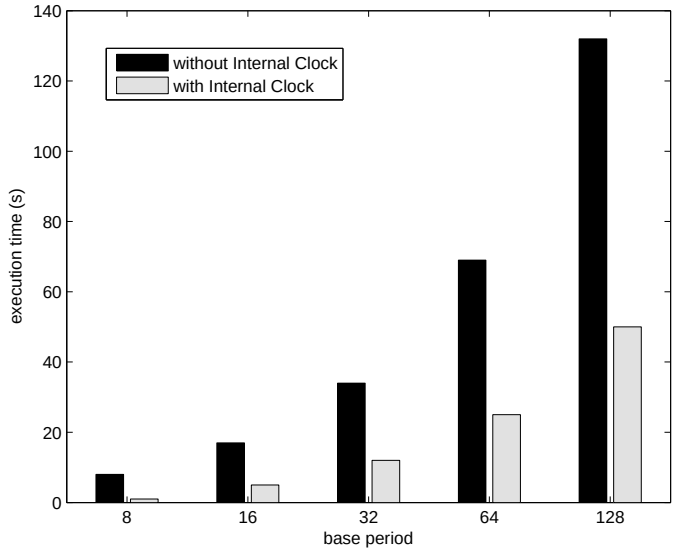


Figure 2.8: Number of behaviors activations with or without adaptive clocks.

We observed a considerable advantage in performance in the case of adaptive clocks. This is illustrated in Fig. 2.8 where we compare the performance in terms of activations for different behaviors (i.e. with different adaptation strategies) considering the overall system. We notice that the adaptive monitoring strategy of `RETURN_TO_NEST` (i.e. with a priori knowledge) produces better results in terms of number of activations.



	with Internal Clock		without Internal Clock	
p_b	number of activations	execution time (s)	number of activations	execution time (s)
8	3	1	8	8
16	4	5	16	17
32	5	12	32	34
64	6	25	64	69
128	7	50	128	132

Figure 2.9: A comparison between the execution of the `RETURN_TO_NEST` behavior with or without the adaptive clock changing the base period p_b.

Then in Fig. 2.9, some results from the analysis of the `RETURN_TO_NEST` behavior are shown, where for each case we plotted the results of one trial. We noted that the number activations of this behavior, with an adaptive clock, was proportional to $log_2(dist)$ whereas, in the case of non adaptive clocks, these increased linearly with time. Furthermore, we noted a significant reduction of the execution time of the behavior endowed with the adaptive clock.

2.5 Test-bed Case Study 2

In this second case study, we present and discuss our framework deployed in different scenarios and settings, both in simulation and in the real world, from simple scenarios to more complex settings. Our aim is to discuss our approach considering its effectiveness efficiency, adaptability, and scalability (considering increasingly complex behaviors and tasks).

For the simulated experiments we used the Stage tool of the Player project [77], while for the real one we used he PIONEER 3DX robotic platform Active Media Robotics, endowed with a blob-finder camera, an odometer and 16 sonar sensors. All the behaviors of the robot are implemented in a cycle using a single thread of execution.

2.5.1 Exploration and prey-predator domain

We evaluated our approach by using a mobile robot that simulates the exploration behavior of a robot endowed with simple visual capabilities. The robot has the task of searching food in its environment while escaping from predators. The robot has to coordinate these activities, by splitting resources among the behavior, taking into account not only the endogenous conditions, but also its internal states. This allows us to associate with this domain different attentional strategies and test them in different levels of complexity.

2.5.2 Attentive Architecture Overview

The robot behavior is obtained as the combination of the following primitive behaviors (see Fig. 2.10): `AVOID`, `WANDER`, `MOVE_TO_FOOD`, and `ESCAPE`.

Behaviors settings and Attentive Updating policies. Below we introduce only the monitoring strategy for the `ESCAPE` behavior, while for the others we refer to the monitoring strategies previously presented. The `ESCAPE` behavior has an internal clock whose frequency depends on the view of a predator. Initially, the base period is set in order to attentively monitor the environment checking for the presence of a predator. The period of this clock depends on the changing of the

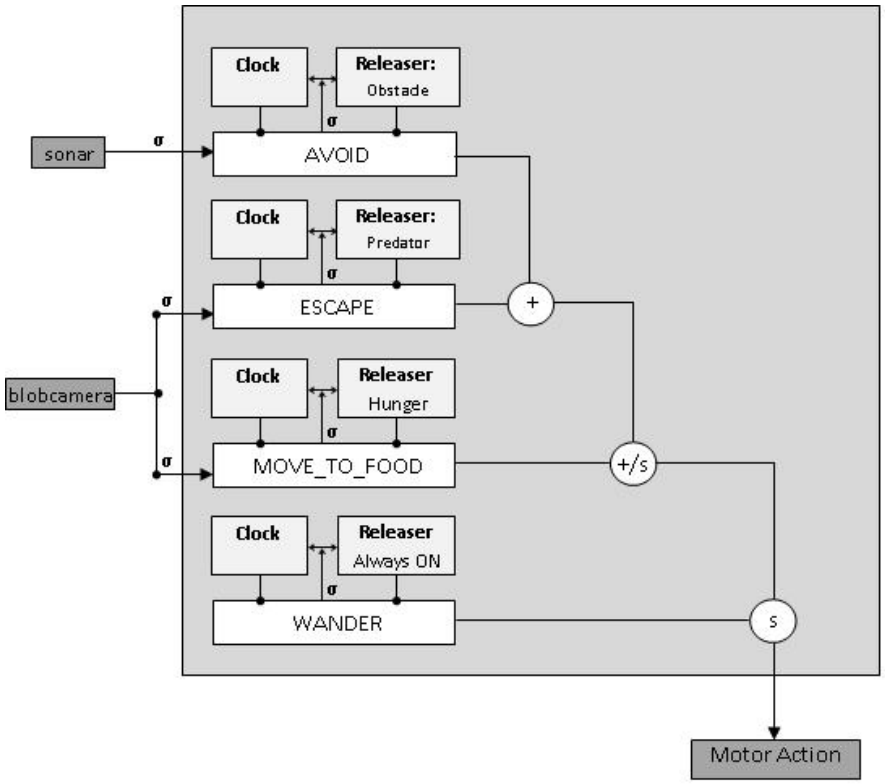


Figure 2.10: Control architecture for the Prey-Predator Domain.

value of the percept itself according to the Weber law:

$$f_a(\sigma(t), p_e^{t-1}) = \frac{\sigma(t) - \sigma(t - p_e^{t-1})}{\sigma(t)}$$

This means that the period will decrease if the predator is moving toward the prey or if the robot is moving toward the predator. So the period will be updated following the (2.3):

$$p_e^t = \rho(t, p_e^{t-1}) * \phi(f_a(\sigma(t), p_e^{t-1}) + (1 - \rho(t, p_e^{t-1})) * p_e^{t-1}$$

The output of this behavior results in a speed deceleration which reflects the state of fear of the robot at the sight of the predator.

System assessment. In these experiments, we evaluated the performance of the AIRM system with respect to the performance of other behavior-based systems without attentional and adaptive mechanisms. In particular, to better assess the gain due to the attentional mechanisms, we compared the system with respect to two different versions of the control system. That is, given the behavior-based architecture depicted in Fig. 2.10, for the comparisons we considered:

(a) a version with *without clocks* (STD) where each behavior can always be activated at each machine cycle, depending on the releasing function (as in a standard IRM-like architecture);

(b) a version *periodic clocks* where each behavior is associated with a clock characterized by periodic, but non-adaptive activations. In this case, we have different clocks without attentional adaptivity.

These two settings allows us to compare the performance of the system with respect to: (a) a *cautious* version of the system, which can monitor and activate each behavior at each control cycle; (b) a *brave* version of the system with the monitoring resolution fixed a-priori (depending on the relevance and criticality of the behavior).

We considered our system working in the prey-predator domain illustrated in the previous section. In this context, we considered different settings and operative scenarios obtained for the combination of the following features: sparsity/density of obstacles (simple/complex), presence/absence of hunger, predator (static/dynamic). Our aim is to assess the system performance by considering: (1) adaptivity in different scenarios; (2) scalability with different behaviors and tasks; (3) effectiveness in terms of tasks accomplishment; (4) efficiency of behavior activations; (5) tradeoff between risk (failures) and opportunity (task accomplishment). To better illustrate the system behavior, we incrementally tested it by evaluating, first, the system performance in relevant subtasks (first scenario) and, then, the emergent behavior of the overall system (second scenario). These experiments are used to evaluate how the emergent behavior scales and changes in different environmental conditions (obstacle configuration), in the presence/absence of conflicting stimuli (predator and food) and internal sources of salience (hunger).

2.5.3 Experimental Results

First Scenario: Incremental test. In the first scenario, the environment is characterized by an area of 20 m x 20 m (400 m^2).

(a) Sparse scenario (b) Dense scenario

Figure 2.11: Map of the environment in two scenarios. Food and predator are, respectively, the red and the blue points. The robot is the blue rounded square.

There are two scenario configurations: with few obstacles (see Fig. 2.11-(a)) or full of obstacles (see Fig. 2.11-(b)). The size of the robot with respect to the environment is 0.2 m ×0.1 m (0.2 m^2). Obstacles are represented by green squares (0.7 m ×0.7 m), while the food by red square in size 0.3 m ×0.3 m.

In this context, we considered the system performance by incrementally adding behaviors and tasks. Initially, we considered a minimal set of behaviors: (Avoiding Obstacles) `AVOID` and `WANDER`. As a second scenario, we considered: (Avoiding Obstacles and Finding Food) `AVOID`, `WANDER`, and `MOVE_TO_FOOD`. For each setting, we collected the data of 10 runs showing the average and standard deviation of the results.

Avoiding Obstacles. In the first set of tests, we consider a robot equipped with the `AVOID` and `WANDER` behaviors, whose task is to safety navigate into the environment with obstacles, for a fixed interval of time (i.e. 5 minutes). This test has been performed in both the sparse and dense scenarios.

In Tab. 2.1, the results of the AIRM system are compared with respect to

	AVOID		no. of danger		speed (m/s)	
S: Sparse / D: Dense	average	st.dev	average	st.dev	average	st.dev
S adaptive clock	403	18	6.8	6.7	0.2874	0.0045
S periodic clock	621	14	24.3	27.4	0.2886	0.0053
S without clock	1203	4	0	0	0.2078	0.0312
D adaptive clock	476	30	3.6	5.5	0.2696	0.0075
D periodic clock	625	3	45.3	49.8	0.2748	0.0136
D without clock	1279	25	0	0	0.1704	0.0118

Table 2.1: AIRM, Periodic and STD architectures endowed with two behaviors and compared in the sparsity and density scenarios.

the caution version (case (a)), *without clocks*, and brave version (case (b)), with *periodical clocks*. The collected parameters are the number of activations of the avoid behavior, the number of possible dangerous situations — minimum distance from the obstacle detected by minimum sonar (less then 0.3 m) — and the average speed of each run.

In Tab. 2.1, we see that both in the case of sparse and that of dense obstacles environment, the number of the different behavior activations is radically reduced in the case of the AIRM architecture. Fewer behavior activations determine a reduction in the computational time spent for sensory data acquisition and processing. This improves the overall system performance in terms of efficient use of resources, since the sensory data are read and processed only when necessary, with a frequency that depends on the environmental circumstances and the internal state of the robot. The average values of all the parameters in the cases of a sparse and a dense scenario for the AIRM case are comparable. These results show that by scaling up the complexity of the environment, we do not lose the benefits of a reduction in the number of behaviors activations and of a high average speed. The results obtained with periodic clocks represent a medium case. Indeed, the periodic setting reduces the behavior activations collected with the setting without clocks; however, without adaptability, we can not ensure robot safety (note the increment of possible dangerous situations in the case of periodic clocks).

Avoiding Obstacles and Finding Food. In the second set of tests, we enhanced the functionality of the system by adding the `MOVE_TO_FOOD` behavior. Here, the task of the robot is to safety navigate into the environment, trying to reach as

much food as possible in a fixed amount of time. The amount of time chosen for the experiments is 3 minutes. As before, we tested the three architectures: with adaptive clock (periodic and adaptive); with periodic clocks (periodic but not adaptive); without clocks (sensory reading at each machine cycle). We tested them both in the sparse and dense environments.

	AVOID		FOOD		WANDER	
S: Sparse / D: Dense	average	st.dev	average	st.dev	average	st.dev
S adaptive clock	310.7	10.4	132.6	67.0	45.2	5.1
S periodic clock	560.2	62.4	17.5	37.0	113.6	22.8
S without clock	968.8	69.8	417.8	197.0	302	101.7
D adaptive clock	330.3	6.8	250.5	114.4	32.6	7.0
D periodic clock	605.2	175.1	28.7	59.4	93.9	29.1
D without clock	1054	43.9	106.5	50.5	408.2	124.9

Table 2.2: Comparing the number of behaviors activations between AIRM, Periodic and Standard architectures in the sparse and dense scenarios.

	no. of danger		speed (m/s)		no. of food	
S: Sparse / D: Dense	average	st.dev	average	st.dev	average	st.dev
S adaptive clock	39.4	25.8	0.282	0.003	1.1	0.6
S periodic clock	136.9	77.1	0.167	0.003	0.2	0.4
S without clock	87.7	40.9	0.175	0.012	1.9	0.7
D adaptive clock	15.2	9.9	0.238	0.018	0.8	0.4
D periodic clock	72	111.4	0.162	0.007	0.4	0.8
D without clock	49.5	15.2	0.169	0.021	1.1	0.6

Table 2.3: Evaluating dangerous situations, medium speed and number of goal reached in comparing AIRM, Periodic and Standard architectures in the sparse and dense scenarios.

In Tab. 2.3, in addition to the parameters presented in the previous tests, we show also the average number of blocks of food found. Contrary to what has previously been observed, the number of `MOVE_TO_FOOD` activations is minimal in the case of a periodic architecture. This fact might suggest to prefer this architecture to AIRM. However, looking at Tab. 2.2, we can see that in the case of a periodic architecture, besides having a decrease in the number of activations, we also have a decrease in the average number of food reached on Tab. 2.3. This happens

because the MOVE_TO_FOOD behavior is responsible of directing the robot toward the food, hence the smaller the number of the activations, the lower the chance of finding food and the precision with which the robot performs the maneuvers during the approach. Moreover, in the periodic architecture, the number of possible dangers grow dramatically, with respect to the AIRM one, where not only the average number of food reached is bigger, but also the conditions of danger decrease. Finally, note that in the periodic case the standard deviation is bigger than the average value itself (for example number of crash and activation of food). This is because the results of the tests present many cases with a zero value and some with a positive number. Now, if we compare the adaptive architecture with the one without clocks, we see not only that the number of activations of behavior is reduced (see Tab. 2.2), but also that, even if in the standard case the average of food found is greater, in the adaptive case the number of possible crashes decreases despite the average speed of the robot remains high (see Tab. 2.3). This means that with the AIRM architecture the robot can reach its goals earlier.

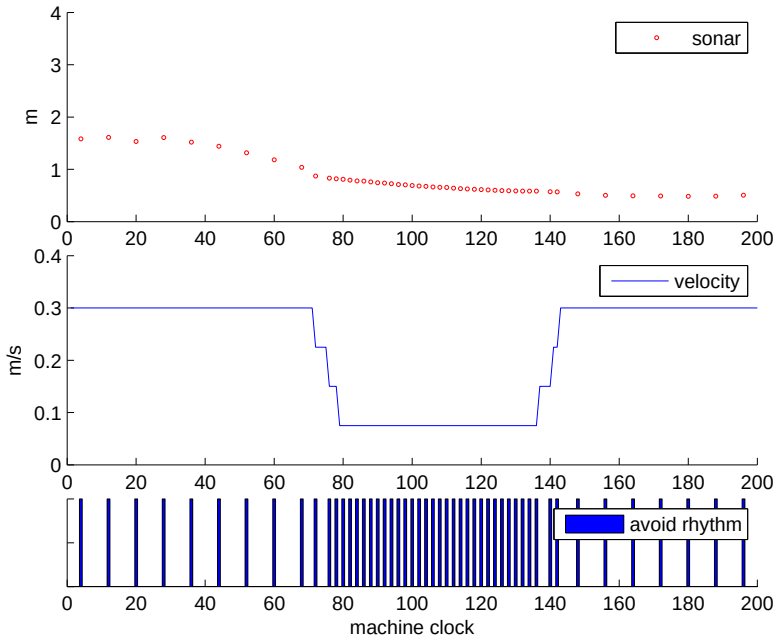


Figure 2.12: AIRM avoidance.

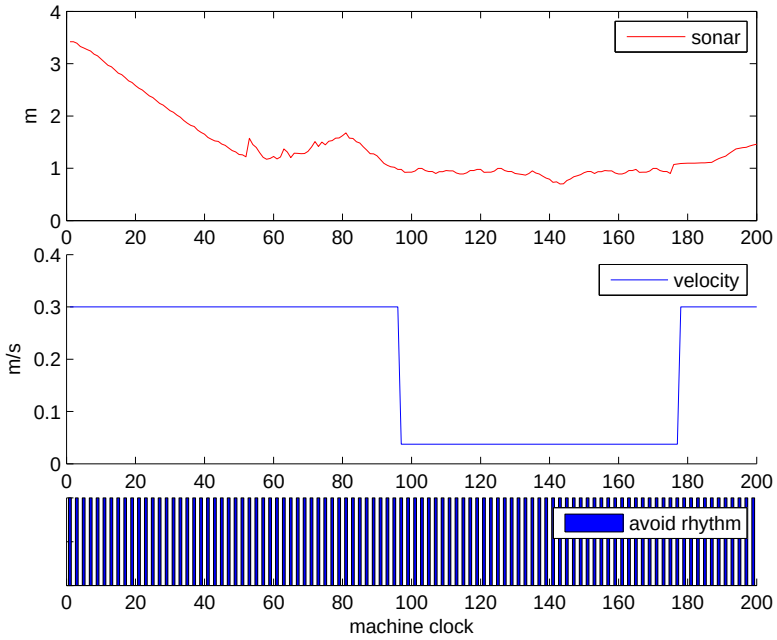


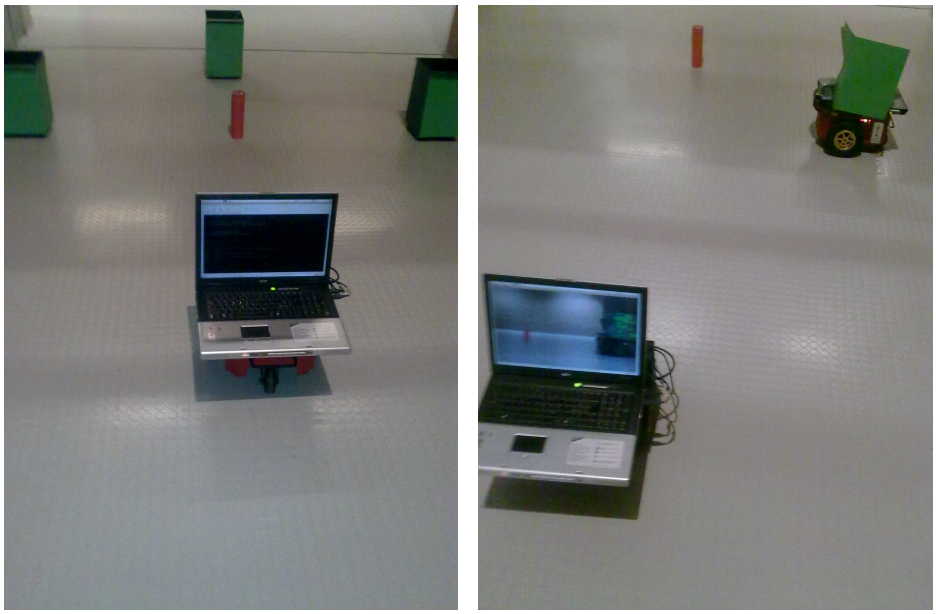
Figure 2.13: Non-adaptive avoidance.

Finally, let us note that the AIRM architecture seems to be still scalable with respect to the changing of the environment (sparse/dense). Moreover, the addition of behaviors makes the AIRM architecture able to select the proper focus of attention on relevant behaviors at each instant of time. This will cause an inversion of trend in the number of possible dangerous situations between the AIRM case and the case without clocks. This happens also because, as we said in the previous section, the `AVOID` behavior is responsible for the speed variations. Indeed it changes the robot speed proportionally to the relevance of the situation. This allows the robot to avoid obstacles in a smooth way (see Fig. 2.12). Indeed, in the non-adaptive case, the speed is very high if there is no danger, very low otherwise; this produces drastic speed variations (see Fig. 2.13).

Second scenario: Testing the system. In a second scenario, the agent is equipped also with an additional behavior: `ESCAPE`, which is responsible for a possible deceleration of speed of the robot according to the presence of one or

more predators. The environment used for this test is free from obstacles. In particular we considered two different settings of the environment for the overall control system test:

- Fixed Multi Predators (static);
- Single Sentinel Moving Predator (dynamic).



(a) Still predators scenarios (b) Single moving predator

Figure 2.14: Map of the environment in two scenarios. Food and predators are, respectively, the red and the green objects.

In both the scenarios the robot has, within a limited amount of time, to reach the food, identified in the scene by a red cylinder, taking into account the presence of the predators. These two experiments have been performed in a real environment. Also for these tests we made 10 runs for each setting.

Fixed Multi Predators. The first setting (static) deals with an environment characterized by the presence of three still predators positioned around the food (see Fig. 2.14-(a)) and we evaluated the emergent behavior of the agent endowed with the AIRM architecture compared with the non-adaptive ones (*without clocks*),

varying the predators distance from the food. In particular the value of the distance vary between a minimum value of 1 meter to a maximum value of 5 meters. In this scenario, the predators are represented by green box of 0.25 m ×0.25 m ×0.35 m, 0.02 m^3, and the food by a red cylinder (0.05 m radius and 0.25 height, 0.002 m^3). We did not evaluate the performance of the periodic architecture because the setting of a proper configuration of fixed periods depends on the initial configuration of predators. Changing such configuration would require also a change in such parameters and the performance will be not be comparable with the others.

	speed AIRM		speed STD		no. of food	
Dist.	average	st.dev	average	st.dev	AIRM	STD
1 m	0.2478	0.0931	0.0114	0.0866	1	0
2 m	0.2679	0.0758	0.0303	0.0494	1	0
3 m	0.2809	0.0601	0.0052	0.0945	1	0
4 m	0.2898	0.0458	0.0071	0.0921	1	0
5 m	0.2934	0.0357	0.0385	0.0106	1	1

Table 2.4: Still predators at different distances.

In Tab. 2.4, we can see the results obtained in comparing the AIRM architecture with the standard one. In the absence of obstacles and being the predators fixed, as expected, the number of possible crashes equals to zero in both cases and were not reported. Since in the standard case the negative contribution to the speed given by the `ESCAPE` behavior weighs on the global motion at each machine cycle, the average speed of the robots is always extremely low. This implies that in most of cases the robot fails to reach food within the target time.

However, in the case of the adaptive speeds, not only the contribution of the `ESCAPE` is subtracted from the overall speed only when the behavior is active, but being the period, and therefore the amount of decrease, proportional to the rate of changes in the environment (in this case the movements of the robot itself towards a predator), the frequency of the behavior activation will tend first to increase and then to relax until it disappears when the robot is close to the food and sees no predators in its visual frame. Indeed, the period depends on the salience of change. As we can see in Fig. 2.15, the first time the robot sees predators, the function of fear ($\frac{\Delta\sigma}{\sigma}$) has a very large peak, leading to a drastic reduction of the period

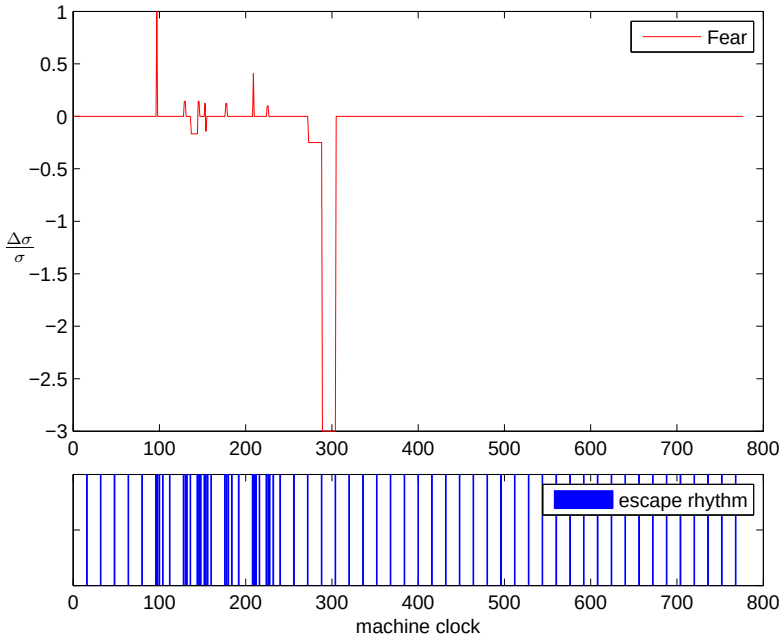


Figure 2.15: ESCAPE frequency with respect to fear.

and thus a relative increase of the frequency of the escape activations. Then fear tends to blur since, being the predators fixed, subsequent readings do not suggest significant changes in scene (i.e. the area of green pixels, which identifies the presence of a predator does not change significantly). For this reason, we see in the plot (see Fig. 2.15) that the frequency tends to relax and then to make some other peak. When the period relaxes too much, but the robot still sense the presence of predators, we must still keep the robot in a *state of alert*— or attention — until the predators go out from the visual frame of the robot and so the function that identifies the fear becomes negative (dangerous situation avoided) and then stabilizes at zero, by relaxing the frequency of activation of the `ESCAPE` behavior until the achievement of the task. This process allows the robot to maintain an high average speed so that, independently of the proximity of predators, it is always able to reach the food within the fixed time.

Moreover, also in this case the adaptive periodicity allows reducing the number of behavior activations, improving the performance of the system with respect to

the standard case, while the adaptability allows exhibiting a flexible and secure behavior, ensuring the robot to safely reach its goal.

Single Sentinel Moving Predator. In the second environmental setting (dynamic), there is only one moving predator (another Pioneer3-dx device covered with a green cardboard), whose task is to move back and forth along a straight direction in order to control the food as a sentinel and to prevent the robot reach the food (see Fig. 2.14-(b)). Here, we evaluate the performance while varying the speed of the sentinel. In particular the sentinel speed vary from a minimum value of 0.15 m/s to a maximum value of 1 m/s, while the robot speed, modulated by the robot behaviors, cannot exceed the maximum value of $0,3$ m/s.

	AIRM				STD			
sentinel	crashes	speed (m/s)		food	crashes	speed(m/s)		food
speed	avg	avg	st.dev	avg	avg	avg	st.dev	avg
1 m/s	0	0.2871	0.0531	1	0	-0.0078	0.1083	0
0,7 m/s	0	0.2751	0.0801	1	0	-0.0103	0.1106	0
0,5 m/s	13	0.2733	0.0870	1	0	-0.0055	0.1060	0
0,3 m/s	0	0.2978	0.0211	1	0	-0.0077	0.1082	0
0,15 m/s	0	0.2976	0.0220	1	0	-0.0040	0.1045	0

Table 2.5: Mobile sentinel mobile with different speeds.

This last set of experiments also highlights how the adaptive architecture has the capability to well distribute priorities among the different behaviors, allowing the robot to reach its goal with a high average speed (see Tab. 2.5) as in the static case. Obviously, the adaptability makes the robot less cautious, leading to the risk of being in possible crash situations. This is not the case of a standard architecture, where the average speed reaches negative values, since the sentinel robot is always in the visual frame of the robot and the `ESCAPE` has the complete priority on the other behaviors for the entire duration of the application.

2.6 Discussion

In this part of the thesis, we investigated the feasibility of the use of adaptive internal clocks to implement an attentional mechanisms of monitoring. We showed the so composed mechanism is able to filter the sensory information and split them among different concurrent/cooperative behaviors, adapting to the surrounding environment changes and to the internal needs of the robot.

While attention-based robot control has been already considered in literature, mainly for vision-based robots, mechanisms for selective attention in robot execution monitoring are less explored. Starting from a behavior-based executive system [79], we introduced simple attentional mechanisms by associating each behavior with an adaptive internal clock that regulates the frequencies of sensor readings and action activations. Here, the process of changing the frequency of sensory readings is interpreted as an increase or decrease of attention towards relevant behaviors and particular aspects of the external environment. In the framework of the schema theory [73], these mechanisms are obtained as a natural extension of IRMs [69]. In this setting, the overall attentional control is an emergent behavior obtained by the interaction of the monitoring strategies.

In particular, such mechanisms can speed up or slow down the period of behavior activation and thereby the reading frequencies of the sensors according to both the robot-environment interaction and the interaction that may arise within the robots itself (its internal states). We not only use a bottom-up selective attention approach to adapt the robot behavior with respect to external events, but we also use the top-down selective attention as a preparatory mechanism. The pre-motor theory of the attention [80] suggests, in fact, that the focus of attention is the consequence also of the act of planning a motor action (e.g. ocular movements). Before that the information to elaborate comes, the attention "prepares" the robot, activating the behavior that will be involved in the elaboration. Hence we use these attentional mechanism in a way that the stimuli in the "attended" position or for the "attended" behavior are recognized quickly and with greater accuracy (because the frequency of reading for that sensor and for that behavior is already high), while the stimuli in the unexpected position are elaborated more slowly (and with little accuracy).

Other authors dealt with flexible/adaptive behavior realized through timed activations. For example, in [53], a parallel architecture focused on the concept of activity level of each schema is presented, which determines the priority of its thread of execution. A more active perceptual schema can process the visual input more quickly and a more active motor schema can send more commands to the motor controller. However, while in our approach such effects are obtained through periodic activation of behaviors, in [53] the variables are elaborated through a fuzzy based command fusion mechanism.

To test the model we implemented a behavior-based control system endowed with these mechanisms and evaluated different monitoring strategies. In particular, we first developed an application which simulates the behavior of the Cataglyphis ant. Furthermore, in order to validate our approach, we also tested the realized control architecture in a prey-predator domain test-bed. We presented a systematic analysis of the attentional system both in a simulated environment and in the real world. In particular, we tested the scalability and the adaptivity of the approach with respect to different and heterogeneous environments and tasks. We evaluated the performance of the attentional system with respect to the performance of other behavior-based systems not provided with attentional and adaptive mechanisms (see Tabs. 2.1, 2.2, 2.3).

The two main advantages introduced by this model of periodic and adaptive innate releasing mechanism (AIRM) observed in all the experiments are as follows:

- Being a mechanism for periodic activation, it enables reducing the number of behavior activations (as opposed to cases where the standard activations are performed every machine cycle), causing a relative decrease of computational load and thus improving the performance of the system;

- The adaptive mechanisms allow the robot to move safe, changing its reaction coherently to the specific environmental conditions. They permit the robot to read sensors more often if there is a dangerous situation and less often in cases of a safe operational situation, showing an “intelligent” behavior.

In our experiments, we observed (see Tabs. 2.1, 2.2, 2.3) that the number of activations of each behavior decreases strongly, i.e. the computational overload is

lowered. Indeed the results obtained by comparing the number of behavior activations in the adaptive architecture with respect to the standard case, show how the adoption of such an architecture actually produces significant improvements of the computation time for sensor processing. It was noted, in fact, not only that the number of activations of the behavior decreases substantially when compared to standards cases where behaviors are continuously activated, but the emergent behavior of the robot remains efficient since its activation is not only periodic, but adaptive with respect to the degree and speed of changes of the surrounding environment. It is to say that (Fig. 2.9) the attentional version of the system reduces the time to achieve the goal (effectiveness) while reducing the activations of each behavior (efficiency). Moreover as you can see in Fig. 2.8 the number of activations is minimum if the period is adaptive and uses an a priori knowledge, is medium if the period is adaptive without a priori knowledge, while the worst among the three, but still better than the case without clock, is the periodic but not adaptive period.

The collected results show also that attentional mechanisms permit a smooth and natural emergent behavior in all the considered scenarios trading off between adaptivity and performance. In fact, if we look at the robot from a behavioral point of view, we observe that in the case of attentional avoid, we have that the robot avoids the obstacle in a smooth way by gradually changing its approaching speed with respect to the proximity of the obstacle (Fig. 2.12), while in the case of a standard application, it reacts faster but less gradually with respect to changes (Fig. 2.13).

Furthermore, the experiments have also shown interesting findings, regarding the timing and scheduling of the behaviors. We know that the monitoring activity is distributed over the concurrent behaviors depending on the frequencies of their associated clocks. In our system, in fact, each behavior is endowed with its own clock, whose period changing is based on external and internal conditions; thus, in a certain instant of the application, each behavior will be characterized by a particular activation period. Therefore it will be activated with higher or lower frequency relating to the value assumed in that moment from this period. Since all the behaviors are combined, intuitively, the behavior with greater frequency will have a greater influence respect to the others in determining the emergent

behavior of the robot. This result can be somehow interpreted as a sort of priority mechanism. This shows a very interesting decentralized synchronization system, that collocates this type of architecture in half the way between a purely reactive and a deliberative system. In fact the clock mechanism, on one hand, allows an immediate response to an external stimulus, as well as in purely reactive architectures; on the other hand, it does so by taking into account the degree of change in the environment and its internal state, thus using the information that contains somehow the "story" of what happened previously, producing, unlike the reactive systems, a non-deterministic response to external stimuli, but a response depending on the particular circumstances and on a previous state. Summarizing, the AIRM architecture permits a smooth and natural emergent behavior that is also more effective (reducing the time to achieve the goal) and efficient (reducing the behavioral activations) with respect to an analogous system without adaptive clocks and it produces a sort of priority scheduling able to organize multi-behavior activities.

In future sections, we will investigate some learning mechanism to select the proper rhythms for each behavior. Moreover, one of the problems of more complex architectures comes from the possibility of arising interferences between different processes. In fact, in our approach, each behavior modulates its own rhythm of activation. Since behaviors may not be independent processes, in the next part we will also move forward in the direction of studying how these adaptive periodical activations of behaviors may influence and constrain one each other.

Chapter 3

Divided Attention Mechanism

3.1 Introduction to Divided Attention

While spatial separation of simultaneous sources of information has been shown to be very effective, little is known about how spatial separation influences performance in tasks in which the subject must pay attention to the content of more than one simultaneous source. In order to achieve this kind of tasks a subject must coordinate his/her activities with the surrounding world, by providing an efficient processing of the large number of stimuli that he receives. These capabilities to process stimuli in parallel and integrate them in a unitary behavior are known in neuroscience as divided attentional mechanism. The divided attention represents a state in which the focus of attention is spread across more than one object or event. We used it in order to make the robot able to integrate in parallel multiple stimuli.

3.2 Motivations

The architecture presented in the previous chapter has provided good results showing, however, that it does not solve the problem of conflicting tasks. Let us know that is possible to have two types of conflicts: a structural conflict or a conflict in resource sharing. In particular, a structural conflict occurs when two tasks use the same channel and they can not be executed at the same time. Else, an interfer-

ence in resources happens when two tasks use the same channel and they can be executed at the same time.

Behavior-based robotics usually resolved conflicts by deploying a subsumption architecture or by implementing some control mechanisms in order to switch between tasks and selecting the action [7], [81] to perform. For example, in [82] the authors presented a schema theoretic model for a praying mantis, whose behaviors are driven by motivational variables such as fear, hunger and sex-drive. In this approach, the action selection module selected only the motivational variable with the highest value. In our approach, the modulation of behavior was not controlled by an on/off switch for changing the task. Indeed, the global behavior emerged from each single behavior through a rhythmic controller modulated by a monitoring strategy. The problem arose when we have to manage concurrent tasks, conflicting in resources sharing. Hence, while we could solve the structural conflict by means of the classical subsumption mechanism, we could not do the same for conflict in resources, since we knew the resources were limited. In this case we needed a mechanism able to opportunely allocate the limited resources in a timely manner between two or more competing tasks. We focused on the divided attentional mechanism. The solution envisaged relied on the usage of the divided attentional mechanisms in order to deal with these conflicting situations. In particular, inspired by study on cognitive distraction, we addressed the problem by introducing mutual influence rules between potentially conflicting behaviors. The human behavior provides several examples of tasks that, while apparently conflicting, are simultaneously carried out. For example, some research analyzed the human behavior while driving and achieving a parallel task [83, 84] (Fig. 3.1). In these experiments the subjects were able to complete tasks in parallel, but the resources allocated to each task must dynamically adapt themselves to environmental conditions and to cognitive and physical capabilities of the subject.

In this section, we will show a divided attentional mechanisms suitable for sensory-motor coordination in the presence of mutually dependent behaviors. We will present our architecture along with a case study where a real robotic system is to manage and harmonize conflicting tasks.

As we mentioned earlier, many of the jobs available in the literature concerning the use of attentional mechanisms, showed how these mechanisms may improve

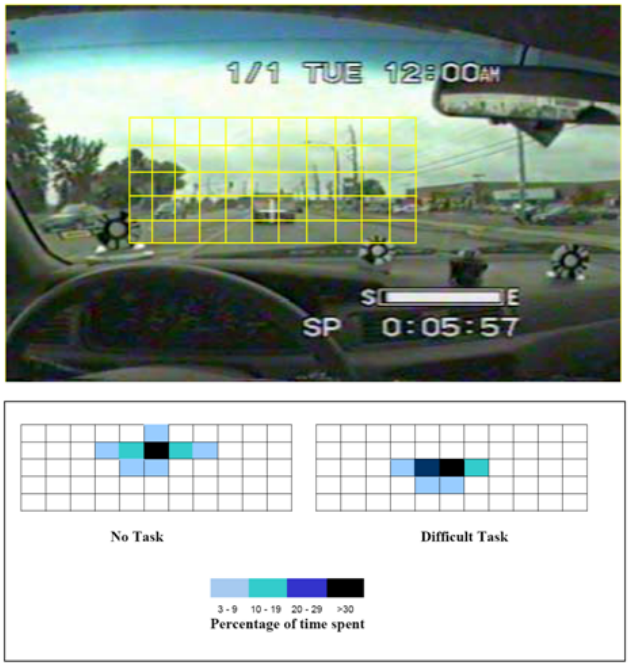


Figure 3.1: Cognitive Load: Driving and Mobile Phones (Harbluk e Ian Noy, 2002 [84]

the robotic vision capabilities. However, attentional mechanisms are necessary not only to focus the attention on salient regions of the space, but also to distribute resources and activities in time [85, 86]. Also in neuroscience, researchers started to investigate the temporal domain of neural activity (for example neural synchrony [64]), and relate such activity to different cognitive processes such as binding, sensory motor-coordination or attentional selection.

With regard to the processes of coordination between competing tasks, von Holst [63] identified the "relative coordination" between endogenous oscillators, which allows an oscillator to influence the frequency of another oscillator. These coordination activities were regarded as "nervous competing activities that do not work either completely independently or through a fixed relationship to one another" (sliding coordinations with phases that deviate or drift away slowly).

Along this direction, we proposed simple mechanisms for distributing the attention on multiple behaviors based on sensory sampling modulation affected by

mutual influence rules, able to regulate dependencies among concurrent conflicting behaviors.

In the AIRMs, each behavior was endowed with an independent regulation mechanism directly depending on internal and external stimuli, and up to date the mutual influence among the parallel behaviors was left as a consequence of the overall self-regulating emergent behavior. However, the notion of divided attention [17] suggests that a limited amount of attention is allocated to tasks, when resources are shared in multi-task behavior, and attention can be available in graded quantity for each task. Indeed, the activation of some behaviors may directly require the activation or the inhibition of other behaviors: two behaviors may not be able to activate themselves as frequently as they need without a degrade of performance (e.g. cognitive load and interference [84]); otherwise, the activation of one behavior may directly induce the activation or synchronization of other behaviors (e.g. synchrony in attentional selection [64]).

Our divided attention general framework was obtained as an extension of the AIRM architecture [87, 68] that integrated mechanisms for mutual influence among attentional behaviors. For this purpose, we introduced simple constraints among the behaviors sampling rates. This mutual influence can work both as an inhibitory or synergic process.

To assess our framework, inspired by the studies on cognitive distraction during driving activities [83, 84], we defined a case study where a real robot is to achieve two conflicting goals. In this context, we compared the performance of this architecture with respect to non-attentional versions of the same system. The empirical evaluation showed that the proposed framework is capable of harmonizing conflicting goals and distracting activities while maintaining an adaptive and reactive behavior.

3.3 Divided Attention Model: Mutual Influence Rules

In the previous sections, we introduced the AIRM (Adaptive Innate Releasing Mechanisms) architecture. In summary, in the AIRM framework, the robotic

system is controlled by a behavior-based executive, where each behavior can be described by a schema theory model [73]. Each behavior is characterized by a Perceptual Schema (PS), which elaborates sensory data, a Motor Schema (MS), producing the pattern of motor actions, and a control mechanism based on a combination of a clock and a releaser. The releaser enables/disables the activation of the MS, according to the sensory data. Instead, the adaptive clock is periodically activated and enables/disables data flow from sensors to PS. When the activation is disabled, sensory data are not processed (yielding to a sensory reading reduction). Furthermore, the clock regulates its reading period p_b (ranging the values in the interval $[p_b min, p_b max]$), hence the frequency of data processing, using a feedback mechanism. Our goal is to develop attentional mechanisms providing a kind of divided attention [17] which focuses sensory resources and modulates task activations by taking into account mutual influences and constraints among the behaviors.

Mutual Influence Rules. In our attentional framework, the attention modulation strategies should be suitably regulated not only with respect to the internal or external saliency, but also with respect to attentional disposition of other behaviors. To account for the problem of mutual influence among attentional behaviors, we propose an extension of the AIRM architecture endowed with explicit constraints among the internal clocks and suitable regulation mechanisms to respect these constraints. The aim is to capture mutual dependencies in terms of interrelations among the clocks' sampling rates and then to regulate the clocks' frequencies according to the presence of conflicting or synergetic behaviors. For example, given two mutually exclusive processes, since these are to be interleaved, the associated clock periods should be opportunely changed to allow their alternated execution; on the other hand, for two concurrent behaviors, the associated clocks are to be aligned: when the frequency of one clock increases/decreases the other clock should be accelerated/decelerated and vice versa. However, we want to add this simple mechanism while maintaining the main features of the AIRM model: the periodic activation of behaviors should provide both a relative decrease in the computational burden and the ability to monitor the internal/external environment.

In this new setting, for each set of clocks $p_1, ..., p_n$, we can introduce a re-

lationship $R(p_1, ..., p_n)$ that specifies the mutual influence. We mainly focus on the relationships between behaviors. In particular, we consider binary constraints $R(p_A, p_B)$ like mutual or synchronized constraints. In this case, the frequencies of the clocks p_A and p_B, associated with the two behaviors, depend not only on the salience of the tasks, but also on the joint frequencies. Examples of these constraints will be provided in the case study presented in the following section.

Related Work. The problem of mutual influence among behaviors were tackled in different approaches. For example, in [88] the author presents a homeostatic system where pairs of behaviors are connected through "successor" or "conflicter links" to inhibit or activate each other. These links play a role which is analogous to that of our mutual constraints; however, our regulation mechanisms are different because they are based on attentional modulation of clocks sampling rates. Moreover, our focus is not on the constraint *per se*, but on the effects of constraints on our architecture.

Concurrent tasks interacting with the attentional processes are considered in [39] where a robot architecture integrates active vision and task execution. However, mutual influence is not considered while attentional and goal-directed behaviors are integrated and coordinated using a perceptual memory.

Our attentional sampling can be also related to flexible scheduling for periodic tasks in real-time systems. In [55], period modulation is exploited only to keep the system load balanced. Similar techniques can be incorporated in our framework; however, in our case, sampling rate and interaction among behaviors depend not only on the computational load, but also on saliencies due to environmental changes, internal states, and goals.

3.4 Test-bed Case Study

The human behavior provides several examples of tasks that, while apparently conflicting, are simultaneously carried out. In many cases, we have not only perceptual or action selection issues, but also cognitive interferences. For example, some research analyzed the human behavior while driving and achieving a parallel task, such as talking over a mobile phone [83, 84]. Driving a car is a complex

behavior that requires the extraction and integration of information from multiple sources. Most of the information relevant for driving are taken by the view, so every change in the visual exploration behavior can be significant for a safe driving. For example, in [84] the authors tried to experimentally assess the effects of cognitive load caused by a secondary task, simultaneously executed. Their results have shown that drivers, under a high cognitive load, execute less saccadic movements consistently with an increase of fixation time and a smaller exploration of the visual field. These experiments show that subjects are able to complete tasks in parallel, but the resources allocated to each task must dynamically adapt themselves to environmental conditions and to cognitive and physical capabilities of the subject.

3.4.1 Conflicting Behaviors Domain

Inspired by these studies [83, 84], we designed a case study with two conflicting goals. In a hallway there are some clusters of green blobs distributed on the left and on the right wall. The robot has the task of running across the hallway in the shortest time possible, while counting all the green blobs (see Fig. 3.3-(a) and 3.3-(b)). The two tasks conflict on the speed of the robot. In fact, the first task would require a high speed, while the second, in order to effectively count all the blobs, would require a slow one.

Environment. The hallway is straight, without obstacles, 14 m long and has a width of 1.60 m (see Fig. 3.2). All along the walls there are 27 green blobs arranged in 3 clusters of 9 blobs each, symmetrically disposed as a 3×3 grid (see Fig. 3.3-(b)). Spots in each grid have a predefined position, with an horizontal distance between spots of 40cm and a vertical distance of 12 cm, while the three grids are randomly distributed along the walls.

3.4.2 Attentive Architecture with mutual influence rules

In order to accomplish the two tasks we implemented three behaviors: `RUN`, `SEARCH` and `SCAN` (see Fig. 3.5).

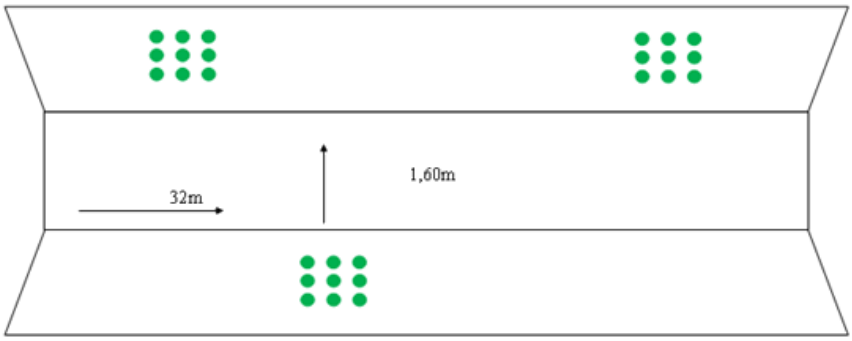


Figure 3.2: A schema of the robot environment.

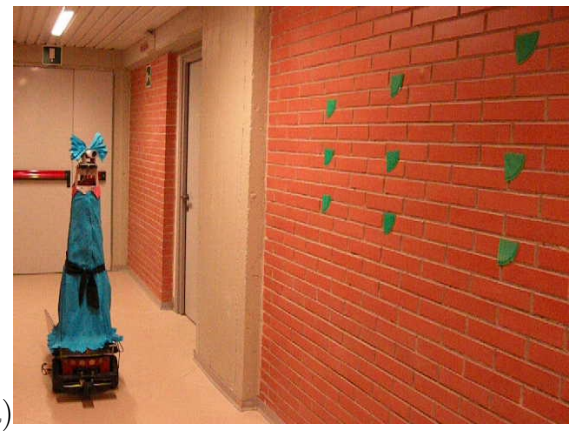
(a)  (b) 

Figure 3.3: (a) A snapshot of the robot in the environment. (b) A snapshot of the robot field of view with a superimposed grid to identify different areas.

The purpose of the `SEARCH` behavior is to search green spots on the left and right wall. In order to accomplish this task, when the behavior is activated, it causes a random movement of the pan-tilt camera. This behavior is activated every machine cycle until is not detected at least one green blob. That is if no green blob has taken over, the clock period of the `SEARCH` behavior is equal to 1 otherwise the period is increased proportionally to the amount of green color detected in the wall (see Fig. 3.4) until a maximum value of 9 machine cycles (i.e. the minimum time to allow to the `SCAN` behavior to identify the 9 blobs composing the set).

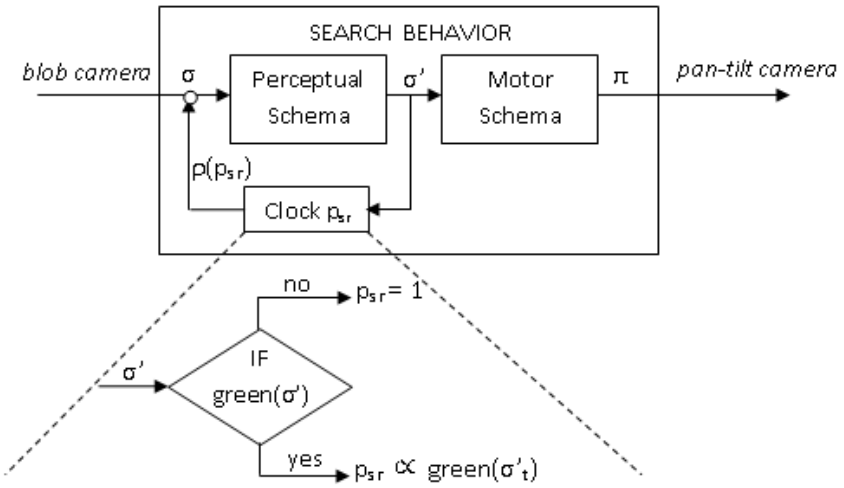


Figure 3.4: Updating of the Search Behavior Clock period.

$$\rho_{Sr} = \begin{cases} \rho_{Sr}/nblob, & \text{if } 1 \leq nblob \leq 9 \\ 1, & \text{if } nblob \leq 0 \\ 9, & \text{if } nblob > 9 \end{cases}$$

The `SCAN` behavior, once that a salient area is identified, has the purpose to count such spots. According to active vision [89], in order to count an object on the wall, the camera has to center the object in its field of view, simulating a saccadic movement. In order to simulate such movements we subdivided the field of view of the camera in nine areas (see Fig. 3.3-(b)). Human way of counting object depends on personal attitudes and may vary among individuals. In our implementation, we realized an algorithms based on the concept of "shortest path". The robot will start to count (and so to center) the top left (or right, according to its direction) spot. After that, the robot will try to center the nearest spot it detects in the peripheral area of view. Let us notice that, in this way, the robot will count first the spots on the same column and then moves to a different row. The clock period of this behavior is modified proportionally to the `SEARCH` behavior activation frequency (see 3.4.2).

In particular the period changes according to the following relation:

$$\rho_{Sc} = \alpha - \rho_{Sr}$$

where

$$\alpha = min(\rho_{Scmin} + \rho_{Srmax}, \rho_{Scmax} + \rho_{Srmin})$$

and $[\rho_{Scmin}, \rho_{Scmax}]$ and $[\rho_{Srmin}, \rho_{Srmax}]$ are respectively the range of allowed values for the `SCAN` and `SEARCH` clock period.

The `RUN` behavior, instead, sets the speed of the robot. Differently from the previous behaviors, the effect of the activation of such behavior continues even if the behavior is off. In fact, after the behavior sent a command to the robot engine, the controller of the robotic system will keep such speed until a new command will arrive. The value of the speed is in inverse proportion with respect to the value of its period. The range of allowed speed is from 0.01 m/s to 0.24 m/s. The clock period of the `RUN` is directly proportional to that of the `SEARCH` behavior (see Section 3.4.2), according to the relation:

$$\rho_R = \rho_{Sr} - \beta$$

where

$$\beta = \rho_{Srmax} - \rho_{Rmax}$$

The system starts with a medium speed, looking for green objects on the walls of the corridor. Its behavior will change according to the visual percept. When the system detects a green object, the `SCAN` behavior period decreases, allowing the robot to slow down its speed and to count the objects it detects. Similarly, if no green objects are detected, `SEARCH` and `RUN` periods become smaller, allowing a more accurate exploration (moving the camera several times right and left, looking for objects), and increasing the system speed in order to reach the end of the corridor as fast as possible.

Mutual Influence Rules. The regulation of mutual influence of two clocks, with periods p_A and p_B, depends on the statical and the dynamical priorities between behaviors, and the relationship $R(p_A, p_B)$.

Relationship If two behaviors A and B, respectively with p_A and p_B periods and with ranges $[p_Amin, p_Amax]$ and $[p_Bmin, p_Bmax]$, share the same resources and are potentially in conflict, we have to define a relationship between these two values. To better understand, we consider what happens in the frequency domain in which a low-pass filter prevents the passage of frequencies below a particular cutoff frequency. If K is this cutoff frequency (i.e. in some way the maximum bandwidth available) representing in our case the maximum rate of behavior activation, and $f_A = \frac{1}{p_A}$ and $f_B = \frac{1}{p_B}$ respectively represent the activation frequencies of two conflicting behaviors, with the relation: $f_A + f_B \leq K$, we indicate that each frequency will benefit from the breadth bandwidth not used by the other and vice versa. Likewise, if the activation period p_A assumes a particular value within its allowed range $[p_Amin, p_Amax]$, the period p_B can only assume a value within $[p_Bmin, p_Bmax]$, limited to the remaining bandwidth. However, if two behaviors need to be executed simultaneously in order to realize a macro behavior, or if their outputs may be summed and are not in conflict, we may assume the following synchrony relationship: $|p_A - p_B| = 0$. In our architecture we have that the `SCAN` behavior (with period p_{Sc}) and the `SEARCH` behavior (with period p_{Sr}) cooperate on the achievement of one of the tasks, but conflicts on the use of the pan/tilt camera. On the contrary, `RUN` (with period p_R) conflicts with `SCAN` on tasks. Indeed, the first has the goal to reach the end of the corridor as soon as possible, while the second needs to slow down as much as possible the speed of robot in order to optimize the counting phase. Finally, the `RUN` behavior and the `SEARCH` behavior can cooperate in the achievement of their own task. In fact, both require a high speed. Let α, β, and γ be constants equal to $\alpha = p_{Sc}max + p_{Sr}min$, $\beta = 0$ and $\gamma = p_{Sc}max + p_Rmin$; in this test the relationships among the periods of these behaviors can be formalized as $\alpha \geq p_{Sc} + p_{Sr}$ (or $\alpha = p_{Sc} + p_{Sr}$ if we want a strong dependence), $\beta = |p_{Sr} - p_R|$ and $\gamma \geq p_{Sc} + p_R$ (or $\alpha = p_{Sc} + p_{Sr}$ if we want a strong dependence).

Priorities Priorities in changing periods depend on the importance of the behavior in accomplishing the task and in ensuring safety of the robot. Behaviors that are safety critical have the maximal priority, hence the other behaviors will be activated consequently. In the case of behaviors with the same priority, the policy for updating the value of the period is "the first takes all", i.e., at each machine

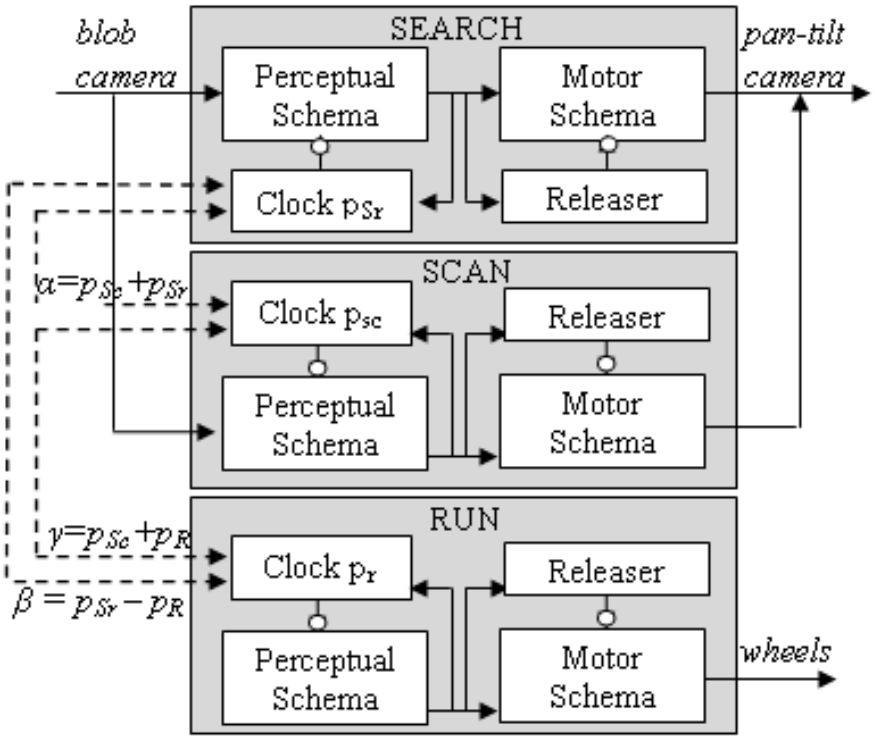


Figure 3.5: Control architecture for the mobile robot.

cycle, the first behavior that changes its period has to notify such variation to other behaviors. The other behaviors have to modulate their periods accordingly. This means that, if a behavior needs a more frequent activation, reacting to what is happening in the environment, a greater set of resources, both computational that sensorial, will be allocated to such behavior. Consequently, others behavior, which in any case keep a periodic activation, will have a smaller number of resources. The period updating policy has to control both the changing in the environment as well as the values of the period of other clocks.

3.4.3 Experimental Results

In order to evaluate the performance of our system, we compared three different architectures, each with different behaviors settings, implemented on a Pioneer 3DX, equipped with a pan/tilt camera and range sonar sensors (see Fig. 3.3-(a)),

and defined as follows:

- **AIRM**: all behaviors are equipped with adaptive clocks;
- **AIRM v max**: adaptive clocks only in `SCAN` and `SEARCH`; the speed of `RUN` is kept constant at the highest value (0.24 m/s);
- **AIRM v med**: adaptive clocks only in `SCAN` and `SEARCH`; the speed of `RUN` is set to a medium value (0.11 m/s);
- S_C2S_R8 **v med**: the behavior activation is periodic (for `SCAN` $p_{S_c} = 2$, for `SEARCH` $p_{S_R} = 8$ and for `RUN` $p_R = 1$), while the speed of the system is kept constant to a medium value (0.11 m/s);
- S_C5S_R5 **v med**: the same as the previous case with different periods (for `SCAN` $p_{S_c} = 5$, for `SEARCH` $p_{S_R} = 5$ and for `RUN` $p_R = 1$);
- **Sub v max**: the behaviors are active at every machine cycle and they are coordinated by a subsumption architecture (i.e., `SCAN` subsumes `SEARCH`). The speed is equal to 0.24 m/s;
- **Sub v med**: the behaviors are active at every machine cycle as in *Sub v max*, but the speed is equal to 0.11 m/s.

In Fig. 3.6-(a), we summarize the results collected during the tests, considering the number of counted blobs and the time spent to complete the task. For each setting, we performed 10 tests. The AIRM architecture performed well in terms of number of blobs counted. In fact, the AIRM implementation counts an average of 17.8 blobs. In the case of the AIRM architecture with adaptive clocks only for `SCAN` and `SEARCH`, the speed of the `RUN` behavior is kept constant during the tests. In these two cases, the number of blobs counted is smaller than the case of AIRM. However, for the AIRM *vmax*, the time performance is better, while in the case of medium speed the average time (127.7 s) is comparable with the AIRM case (123 s). Another important thing to highlight is that, while the average speeds in the medium case and in the AIRM *vmed* case are comparable, the number of blobs counted is better in the AIRM case. This is because the system will adapt itself

(a)

	no. of blobs	time (s)
AIRM	17.8 ± 3.22	123 ± 11.25
AIRM sp. med	7.6 ± 2.76	127.7 ± 1.64
AIRM sp. max	5.9 ± 1.45	60.4 ± 1.84
S_C5S_R5 sp. med	4.6 ± 1.07	127.5 ± 1.96
S_C2S_R8 sp. med	9.9 ± 2.08	128.7 ± 3.37
Sub sp. med	13.8 ± 2.49	127.4 ± 2.01
Sub sp. max	7.4 ± 1.17	61.7 ± 2.83

(b)

	error
AIRM	0.07
AIRM sp. med	0.15
AIRM sp. max	0.35
S_C5S_R5 sp. med	0.18
S_C2S_R8 sp. med	0.13
Sub sp. med	0.10
Sub sp. max	0.32

(c)

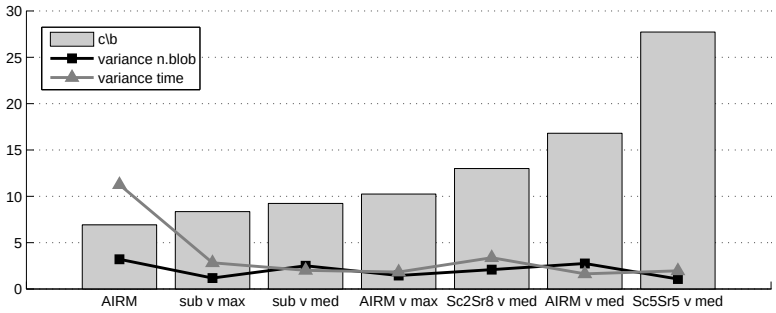


Figure 3.6: (a) Performance and standard deviations in term of number of counted blobs and time spent to accomplish the task. (b) Error on the number of counted blobs for units of time. (c) Plot of costs/benefits of the tests. C/b is evaluated as time/counted blobs.

to the surrounding environment speeding up or slowing down, taking advantages of empty areas to accelerate, while decelerating when it perceives blobs to count.

In the cases of periodic (not adaptive) activation of behaviors (S_C5S_R5 and S_C2S_R8), the performance with respect to the number of counted blobs is worst than in the AIRM case. The case S_C5S_R5 presents the worst results in terms of counted blobs. We experienced a little improvement in the case of more frequent activation of `SCAN` (S_C2S_R8). However, we have to highlight that the periodic

activation of behaviors in the case of $S_C 2 S_R 8$ determines a higher number of activation of perceptual schemas (i.e. wasting more resources) with respect to the AIRM case, elaborating camera data even during the exploration of empty areas.

In the last set of tests, we evaluated the performance of a subsumption architecture (*Sub*). In this implementation `SCAN` subsumes `SEARCH`. The speed of `RUN` is kept constant at 0.24 m/s and 0.11 m/s. The performance in *Sub v med* is better than the other cases except for the AIRM that performs the best. Indeed, the subsumption architecture resolves potential conflicts on resources (i.e. the pan/tilt of the camera) while, without an arbitrator module, such conflicts may degrade the performance. However, in this case, analogously to the periodic activation of behaviors, we have a higher number of activations of the `SCAN` perceptual schema that elaborates camera data at each machine cycle. These results make us foresee that, in the case of a higher elaboration load, an adaptive architecture may significantly improve the performance.

In Fig. 3.6-(c), we plotted the cost/benefit (time/counted blobs) evaluation. Also, from this point of view, the AIRM implementation performs better than the others. However, this plot shows that the AIRM case presents a greater standard deviation in the time performance. A high standard deviation implies a high variability of the test results. This variability is caused by the adaptability of the system with respect to the environment and, consequently, to the changes of the system speed.

Finally, in Fig. 3.6-(b) we evaluated the error on the number of counted blobs for units of time. This error is evaluated as $(nB - nCB)/t$, where nCB is the number of counted blobs, nB is the total number of blobs in the environment and t is the time spent to accomplish the task.

3.5 Conclusions

We investigated simple attentional mechanisms for coordinating competitive and cooperative behaviors in a behavior-based robotic system. We showed the so composed mechanism is able to opportunely split resources among different concurrent/cooperative behaviors. The results show that the AIRM mechanisms are effective in adapting the frequency of behavior activations according to the particular cir-

cumstances, incrementing or decreasing the attention towards salient aspects of the robot environment or the internal state and incrementing or decrementing the related behaviors by means of suitable mutual influence rules. We compared our architecture with different architectures not endowed with attentional mechanisms. In summary, we observe that the proposed architecture performs better than the others in terms of: number of detected blobs (effectiveness); tradeoff between time and counted blobs (cost/benefit); error of detection (precision); fewer activations of the perceptual schema (efficiency). Basically, the system can modulate the activation frequencies on the basis of the available resources and external conditions. Indeed, by using the adaptive clocks, the number of behaviors activations substantially decreases compared to the case where the control system enables the robot behaviors at each machine cycle, and this results in a substantial gain in performance.

Concurrent tasks interacting with the attentional processes are considered in [39] where we find a robot architecture integrating active vision and tasks execution. However, here mutual influence is not considered, while the attentional and goal-directed behaviors are integrated and coordinated using a perceptual memory. Our attentional sampling can also be related to flexible scheduling for periodic tasks in real-time systems [54, 55]. Here, analogously to our system, period modulation is exploited to degrade computation and keep balanced the system load. For example, in [54] the authors propose an elastic model to decide how to change the sampling period associated with a task. Similar techniques can be incorporated in our framework; however, in our case the sampling rate depends not only on the computational load, but also on the saliency due to environmental changes, motivations, and goals.

Part III

Learning the Attentive Monitoring Strategies

Chapter 4

Learning the Attentive Strategies

4.1 Introduction

In this section, an evolutionary process to regulate the attentional executive control of the AIRM architecture is presented. In particular, we deployed a Differential Evolutionary algorithm [90] to tune a set of parameters encoding the agent attentional strategies. We evaluated the approach in an adaptation and survival scenario. In this context, we observed that the evolutionary process provides interesting solutions after few iterations. To validate our approach, we experimented the generated control system in different environments. The collected results showed that the generated settings are more effective than the hand-tuned ones. Furthermore, we observed that the generated control systems remain effective when the environmental conditions are changed.

4.2 Motivations

Similarly to the concept of adaptability, the understanding of the learning processes is important for cognitive robotics because these processes underly the development of cognitive ability. Hence one natural extension of the system has been the introduction of a learning mechanism able to select the appropriate monitoring strategy for each behavior, starting from some specific parameters. Indeed, despite the results obtained by different implementations of the AIRM architecture

were very good when compared to the standard case (the one that is continuously monitoring the environment), if we radically changed the environment, the system did not show the desired behavior. So we have thought to modify the updating functions of the rhythms by adding some "attenuation parameters" that can adjust the trend of functions on the basis of the particular experimental conditions (type of environment, physical limitations of robot, considerations about the task to be pursued, etc.) and using a learning algorithm in order to determine such context-dependent parameters with the aim to make the system general-purpose.

4.3 Learning with a modified Differential Evolutionary Algorithm

In order to set the attentional parameters for the updating functions, we introduced an evolutionary method. More specifically, we deployed the *Differential Evolution* (DE) *Algorithm* to choose, in the space of possible solutions, the continuous parameters that best regulate the clock sampling rates of the behaviors. Among different evolutionary algorithms, such as *Genetic Algorithms* or *Particle swarm optimization*, we considered DE, since it is one of the best algorithms in the literature that allows exploring a range of continuous values that is not restricted (see [90]).

DE works as a genetic algorithm that gradually build the robot control system by optimizing the problem of maintaining a population of candidate solutions and developing a new candidate solution, by combining the existing ones, according to the classic crossover and mutation operators, i.e. maintaining the most suitable solution for the optimization problem.

The approach is to generate an initial population of NP individuals $x_i^k = (x_{i,1}^k, \ldots, x_{i,D}^k)$, randomly different from each other, where D is the number of the parameters to be learned, $k = 0, \ldots, GEN$ is the generation and $i = 0, \ldots, NP$ is the member of the population. Since the D attenuation parameters to be learned are responsible for changing the attentive monitoring strategies, they affect the global control system of each member. So the performance of each member depends on the combination of these parameters which are evaluated by using the variations

present within a population of solutions during the process of interaction between the agent and its environment. The general formulation of the problem is to consider an *objective* or *fitness function*, that evaluates the performance of the system depending on the choice of the attenuation parameters used for regulating the attentive monitoring strategies, and to solve the minimization problem by finding the parameters combination (or gene combination) that produce the best (or minimum) value for the fitness.

DE works generating a new individual from each existing individual i using one fixed (usually the best) and two random individuals. In particular, in each generation the best existing individual x_b^k in the population is determined. For each individual x_i^k in population, the difference between the best individual and the selected individual is computed. Then two random individuals $x_{r_1}^k$ and $x_{r_2}^k$ (different from best and i-th individuals) are selected and the difference between them is also computed. A portion of these two differences is added to the i-th individual and form the new individual x_i^{k+1}. Therefore the new individual is calculated as:

$$x_i^{k+1} = x_i^k + F * (x_b^k - x_i^k) + F * (x_{r_1}^k - x_{r_2}^k) \tag{4.1}$$

where $F \in [0, 2]$ is called the *differential weight* or also *mutation factor* and represents a constant coefficient controlling the amplification of the two differences.

In order to impose more diversity to the new generation, a crossover operator CR is introduced. For each gene $x_{i,j}^{k+1}$ of new individual x_i^{k+1}, a random value $rand_{i,j} \in [0, 1]$ is chosen and compared with the probability of crossover CR. If $rand_{i,j}$ is less than the crossover probability, the newly generated gene is retained; otherwise the new gene is copied from the predetermined old individual

$$x_{i,j}^{k+1} = \begin{cases} x_{i,j}^{k+1}, & \text{if } rand_{i,j} \geq CR \\ x_{i,j}^k, & \text{otherwise} \end{cases} \tag{4.2}$$

The last step in DE algorithm is the comparison of the newly generated individual x_i^{k+1} with the old individual x_i^k.

$$x_i^{k+1} = \begin{cases} x_i^{k+1}, & \text{if } fitn(x_i^{k+1}) \leq fitn(x_i^k) \\ x_i^k, & \text{otherwise} \end{cases} \tag{4.3}$$

Once that its fitness value is calculated, if the resulting individual yields a better fitness value than the predetermined individual, the newly generated individual replaces the old individual; otherwise, the old individual is preserved.

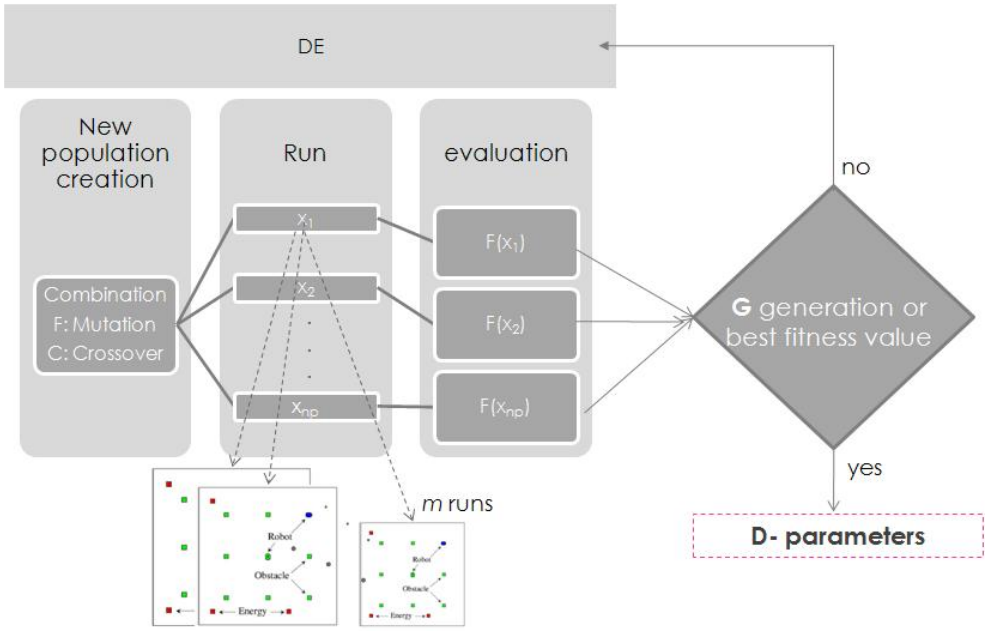


Figure 4.1: Differential Evolution Schema.

In Fig. 4.1 a schema representing how the DE algorithm works is shown. Summarizing, the algorithm starts initializing the starting D parameters with those of the manual tuning. At each generation it generates a new population of Np individuals from the existing ones using the crossover and mutation factors and tests the behavior of each member of the new population by launching a run of the application in the considered case study domain. Then, once evaluated the fitness function corresponding to each individual, the algorithm seeks for the best individual in order to create the new generation. The process is repeated for G generation or until an suitable fitness value is reached. The only difference of our implementation of the DE algorithm with respect to the existing one is that since the specified environment is not deterministic, the same parameter combination can result in different fitness values. Hence, for each individual, we calculate the average of the fitness values evaluated on m = 10 runs.

4.4 Test-bed Case Study

Based on the model introduced in the previous sections, we designed a behavior-based robotic system which use different attentional monitoring strategies to regulate the partition of resources distributed among the different behaviors, by endowing the updating function with attenuation parameters able to adjust the function trend.

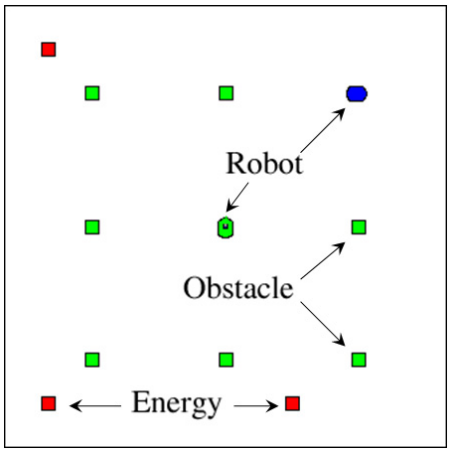


Figure 4.2: Adaptation and Survival Domain.

Platform. As well as in other experiments we used a simulation of a PIONEER 3DX provided with a blob camera, an odometer sensor, and 16 sonar sensors. The robot is controlled by a Player/Stage client [77].

4.4.1 Adaptation and Survival Domain

The robot (see Fig. 4.2) is to explore a dynamic and unknown environment, avoiding obstacles and seeking a source of energy to recharge its batteries when necessary. The proximity of other robots may cause a defensive attitude of our robot. Indeed, in this case, it will tend to back off to avoid possible collisions. This domain is extremely complex since it is tied by the combination of different strategies: low level activities such as avoidance, high level tasks such as searching and reaching of some target, and some other possible conflicting task such as escape from a danger. Our system has to control and coordinate these activities, splitting

resources among behaviors, directing the attention of the robot towards salient perceptual stimuli.

Behavior-based control. The robot behavior is obtained as the combination of the following primitive behaviors: `AVOID`, `SEARCH_BATTERY`, `MOVE_TO_BATTERY`, and `HESITATE` (see Fig. 4.3).

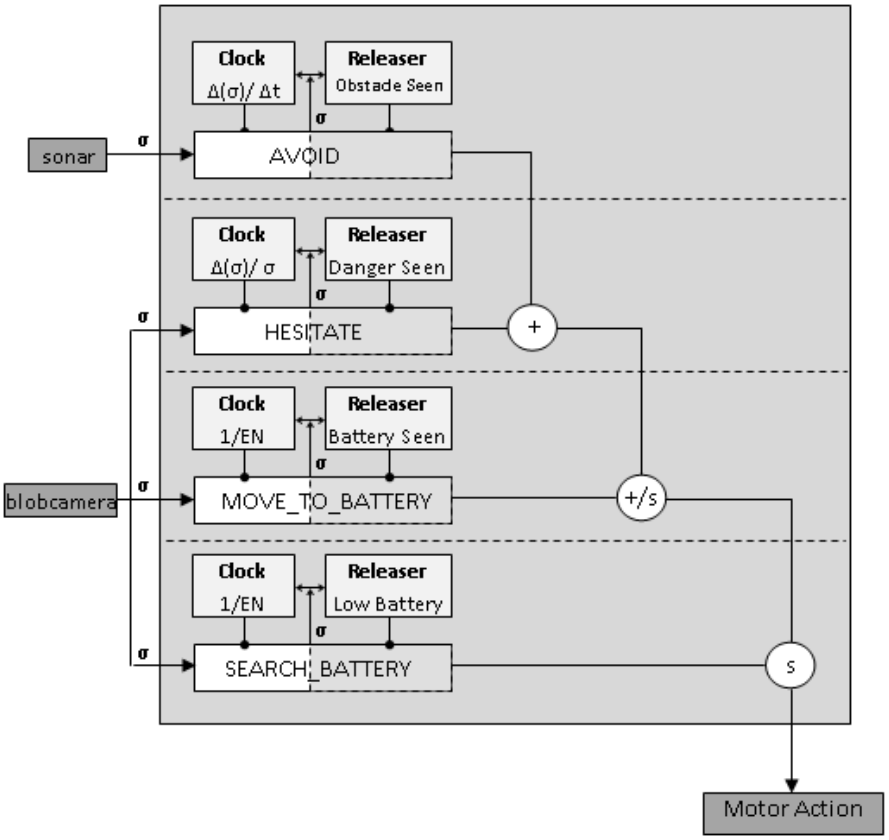


Figure 4.3: Control architecture.

4.4.2 Attentive Architecture Overview

The `AVOID` behavior is responsible for obstacle avoidance. This behavior is safety critical and needs an updating policy for its adaptive clock which is able to timely react to dangerous situations. In this case, the focus of attention for the `AVOID` clock period can be inversely associated to the input percept rate of variation. More

formally, the period p_b^t is updated through the following updating focus function:

$$f_a(\sigma(t), p_a^{t-1}) = \alpha_{avoid} * \left(\frac{1}{RATE + k_{avoid}}\right)$$

where, the $RATE$ parameter identifies the first derivative of sensors signal with respect to time:

$$RATE = \frac{\Delta\sigma(t)}{\Delta(t)}$$

In particular $\Delta(t)$ is equal to p_a^{t-1}, that is the period at the previous clock cycle, $\Delta\sigma(t)$ is equal to $\sigma(t) - \sigma(t - p_a^{t-1})$ that is the difference between the actual data perceived by the sonar sensor $\sigma(t)$ and the data received at the previous sampling step $\sigma(t - p_a^{t-1})$. In this way, the `AVOID` activations frequency adapts itself not only to the environmental changes, but also to the speed at which these changes take place. Also, α_{avoid} and k_{avoid} are two attenuation parameters, useful to smooth the evolution of the function which defines the period. These two parameters are context dependent and will be tuned by the DE algorithm.

The distraction mechanism can be implemented with a linear function, depending from a β_{avoid} attenuation parameter:

$$f_d(p_a^{t-1}) = p_a^{t-1} + \beta_{avoid}$$

The `SEARCH_BATTERY` behavior provides a random search of source of energy in the environment. The frequency of this behavior activation is related to the charge level of the robot battery, that is a linear time-dependent function that represents the agent need of energy. This means that at the beginning, when the value of the need of energy is low, the `SEARCH_BATTERY` behavior is released with a predefined period that depends on the life cycle of the agent. Then, the lower the battery, the greater the need for the robot to turn around in order to look for it. The energy need is represented by a parameter EN that increases linearly with time, and the update function of the period will be expressed as follows:

$$f_a(\sigma(t), p_s^{t-1}) = \frac{k_{search}}{EN} + h_{search}$$

where

$$k_{search} = \frac{(MAX\_search\_period - 1) * MIN\_EN * MAX\_EN)}{(MAX\_EN - MIN\_EN)}$$

and

$$h_{search} = \frac{(MAX\_EN - MAX\_search\_period * MIN\_EN)}{(MAX\_EN - MIN\_EN)}$$

are attenuation parameters, depending on the $MAX\_search\_period = p_s max$, that will be set by the DE algorithm; MIN_EN and MAX_EN are constants depending on the life cycle of the robot. The output of this behavior is a random pattern of orientations for the motor action.

The `MOVE_TO_BATTERY` behavior guides the agent towards the battery once it has been identified. So the releaser is activated by battery detection using the blob camera. The period of this behavior activation also depends on the same parameters regulating the `SEARCH_BATTERY` behavior. In particular, if the clock is on and the agent sees the battery, the output will be a movement towards it, otherwise the agent will rely on the `SEARCH_BATTERY` behavior.

The `HESITATE` behavior has an internal clock whose frequency depends on the view of a possible danger (an unknown moving object or an object of a particular color, different from the battery, etc.). Initially, the base period is set to safely monitor the environment checking for the presence of some danger. If the robot senses an unknown object of blue color (for example another robot), and the behavior is enabled, the robot turns in the opposite direction of the danger. This means that the period will decrease if the robot is moving towards the danger:

$$f_a(\sigma(t), p_h^{t-1}) = \alpha_{hesitate} * \Big(\frac{1}{FEAR + k_{hesitate}}\Big)$$

where, the $FEAR$ parameter identifies the degree of fear of the robot, calculated with respect to the proximity of a dangerous object according to the *Weber law of perception*:

$$FEAR = \frac{\Delta\sigma(t)}{\sigma(t)}$$

whit $\sigma(t)$ referring to the camera percept; also $\Delta\sigma(t)$ is always equal to $\sigma(t) - \sigma(t - p_h^{t-1})$, $\alpha_{hesitate}$ and $k_{hesitate}$ will be set by the DE algorithm.

When the robot does not perceive the blue object, its clock period is relaxed according to the following linear function:

$$f_d(p_h^{t-1}) = p_h^{t-1} + \beta_{hesitate}$$

4.4.3 Experimental Results

In the case study, we aim at deploying a *Differential Evolution (DE)* [90] algorithm in order to learn the parameters associated with the attentional strategies. In our case each member of a generation is characterized by the combination of the following parameters: α_{avoid}, k_{avoid}, β_{avoid}, $\alpha_{hesitate}$, $k_{hesitate}$, $\beta_{hesitate}$ and MAX_search_period.

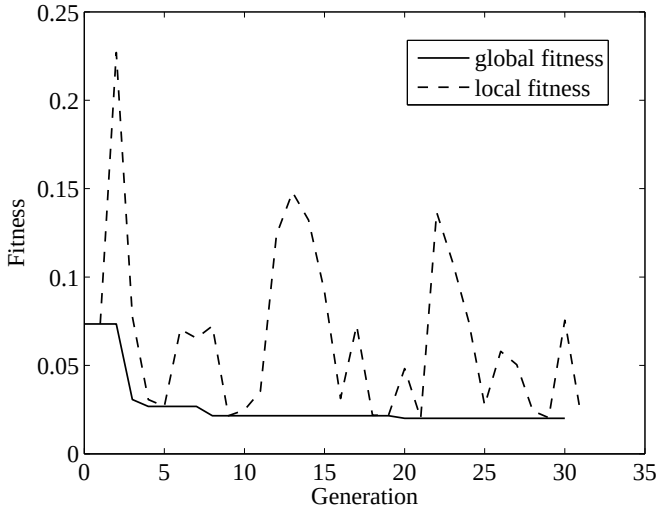


Figure 4.4: Fitness Evolution.

In order to evaluate the performance of each individual, we introduce the following fitness function:

$$\begin{aligned} fitn = \quad & M_1 * (1 - e\_f) + M_2 * \tfrac{num\_d}{m\_c} + M_3 * \tfrac{t_{tot}}{t_{max}} + \\ & M_4 * \tfrac{c_a + c_s + c_m + c_h}{m\_c * num\_b} + M_5 * coll + \\ & M_6 * \tfrac{s_{max} - s_{avg}}{s_{max}} \end{aligned}$$

where we evaluate, respectively, whether the energy source has been reached (e_f), the degree of reliability in terms of the percentage of dangerous situation (num_d) per machine cycle (m_c), the time spent to accomplish the task (t_{tot}) respect to the max allowed time (t_{max}) for each run, the percentage of behaviors activations (c_a, c_s, c_m, c_h) per machine cycle and with respect to the total number of behaviors (num_b), the number of collisions with a moving object ($coll$) and the average speed (s_{avg}) with respect to the maximal (s_{max}). Fitness values are in the range $[0, 1]$, where 1 is the worst result and 0 is the optimum, and $M_1, \ldots, M_6$ are constant weights. Hence, the fitness is a value that we want to minimize by opportunely tuning the attenuation parameters in order to balance the tradeoff among these performance measures. Following the DE algorithm described before, we start producing an initial generation G_0, of $NP = 20$ individuals, by randomly choosing real values in an unbounded space for the considered parameters. For each individual we launched a $m = 10$ times simulated experiments, collecting the average fitness values. At the end of the $NP * m$ simulations the algorithm selects the best fitness value for the global experimentation and a local best fitness value for the current generation. This process is repeated for $GEN = 30$ generation.

Figure 4.4 shows the evolution of the global best fitness (solid line) and the local best fitness (dashed line) obtained at each generation. Notice that the best fitness starts decreasing after few generations.

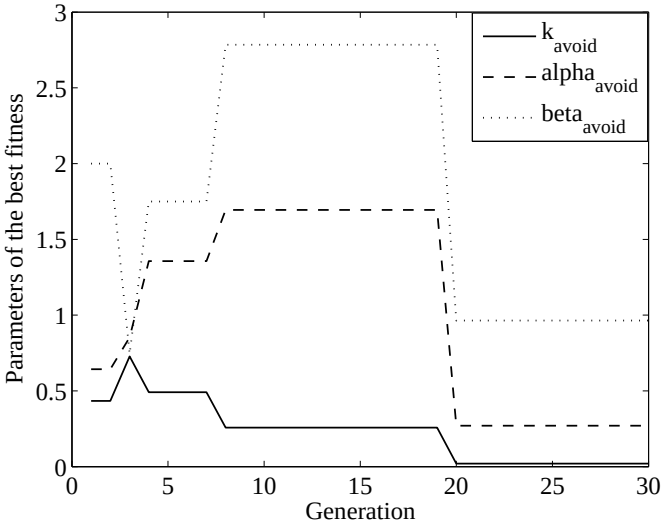


Figure 4.5: Evolution of the attentional `AVOID` parameters.

In Fig. 4.5, 4.6 and 4.7 we show the evolution of the attentional parameters values respectively for `AVOID`, the `HESITATE` and the `SEARCH_BATTERY` behaviors. Let us notice that, following the DE algorithm, the parameters explore a wide range of values.

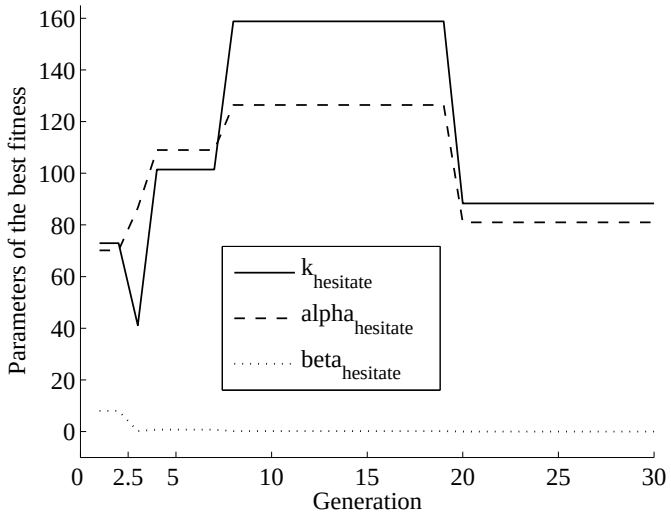


Figure 4.6: Evolution of the attentional `HESITATE` parameters.

The evolutive algorithm highlighted an interesting discrepancy between the updating strategy for `HESITATE` and `AVOID` with respect to the β parameters (dotted lines). For both these behaviors, the distraction function is a linear function that differs only on the β parameter. In the `AVOID` case the DE best value is $\beta_{avoid} = 0.96$, while for the `HESITATE` we have that $\beta_{hesitate} = 0.04$, that is a much slower decrease of the period. These results may be motivated by the fact that, while `AVOID` has to deal with static objects (i.e. once that an obstacle is avoided the robot can freely relax its period), `HESITATE` has to manage the interaction with a moving object (i.e. the avoiding process depends on the movements of both the robot and the object). This sort of cautiousness also depends on the fact that in the test environment the probability to meet again the moving robots is high. Moreover, the magnitude of the values of α (dotted lines) and k (solid lines) parameters in both `HESITATE` ($\alpha_{hesitate} = 81$, $k_{hesitate} = 88$) and `AVOID` ($\alpha_{avoid} = 0.27$ and $k_{avoid} = 0.02$) is directly related to the magnitude of their percepts, namely the speed and the blob area.

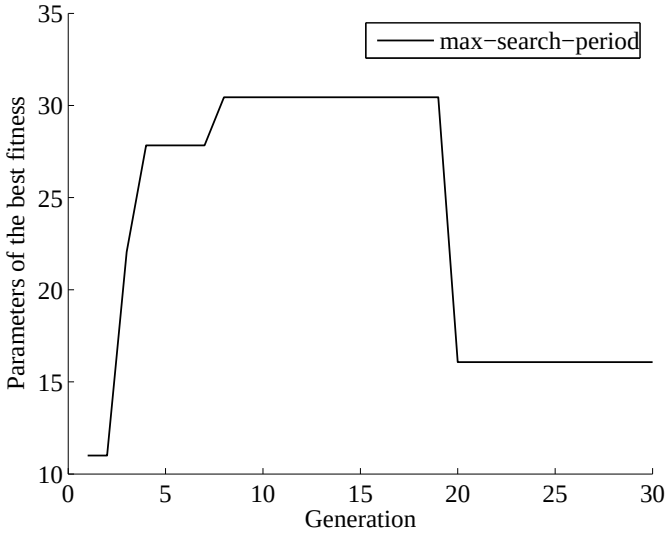


Figure 4.7: Evolution of the attentional SEARCH_BATTERY parameters.

Finally, let us notice that the value of MAX_search_period, after trying bigger values in order to minimize the number of activations of the SEARCH_BATTERY behavior, converges to a smaller value (MAX_search_period=16). This fact may be motivated by the fixed amount of time for each simulation and by the fact that the sources of energy are few with respect to the simulated environment, and not directly accessible.

4.4.4 Validation Tests and Conclusion

In Tab. 4.1 and 4.2 we show the results obtained in a set of validation tests. First of all, we compare the results of the global performance (calculated in terms of fitness value, number of dangerous situation and number of behavior activations) of the system endowed with the best DE parameters (first row) with respect to those manually tuned at a first implementation of the AIRM architecture (second row). We note that the performance drastically improves from the manual case to the learned one (see the fitness value in the first column, while comparing the first and second rows in Tab. 4.1).

Indeed, the value of the manual case fitness (0.147) is a magnitude order larger than that of the test case (0.025). The values of parameters obtained by the DE

	Fitness		Dangers	
Environments	average	devstd	average	devstd
Env Test	0.025	0.007	4.8	11.2
Env Test Manual	0.147	0.150	34.6	23
Env Similar	0.027	0.017	4.6	14.5
Env Complex	0.051	0.029	20.8	40.7

Table 4.1: Experimental results.

	Avoid Call		Hesitate Call		Search Call	
Environments	average	devstd	average	devstd	average	devstd
Env Test	14.6	3.9	18.8	9.5	17.8	12.1
Env Test Manual	51.1	21.4	18.7	3.8	16.1	9.8
Env Similar	11.8	2.8	14.5	2.1	6	1.5
Env Complex	35.1	32.4	136	134.6	37.6	32.1

Table 4.2: Behavior activations.

algorithms for the `AVOID` and `HESITATE` behaviors are completely different from the manual setting ($k_{avoid} = 1, \alpha_{avoid} = 0.8, \beta_{avoid} = 2, k_{hesitate} = 1, \alpha_{hesitate} = 1, \beta_{hesitate} = 2$). The only value equal for both the manual and the DE is the MAX_search_period.

Moreover, in order to validate the learned parameters and test the adaptability of the associated system, we assessed the performance in two new environments: (i) similar to the training one in the number of obstacles and energies (see Fig. 4.4.4-(a)); (ii) more cluttered then the training environment (more obstacles and traps) (see Fig. 4.4.4-(b)).

The collected results are depicted in Tab. 4.1 and 4.2 in the second and third columns. We notice that, in the case of a similar environment, the global performance does not differ very much from the test environment both in terms of fitness values (Tab. 4.1), behavior activations and number of dangerous situations (Tab. 4.2). This is to say that the attentional system, endowed with parameters learned by DE technique, is robust, since it seems to remains stable in similar environment. By making the environment more complex, we note, as expected, a little degrade in performance (see for example the fitness value = 0.051, or the

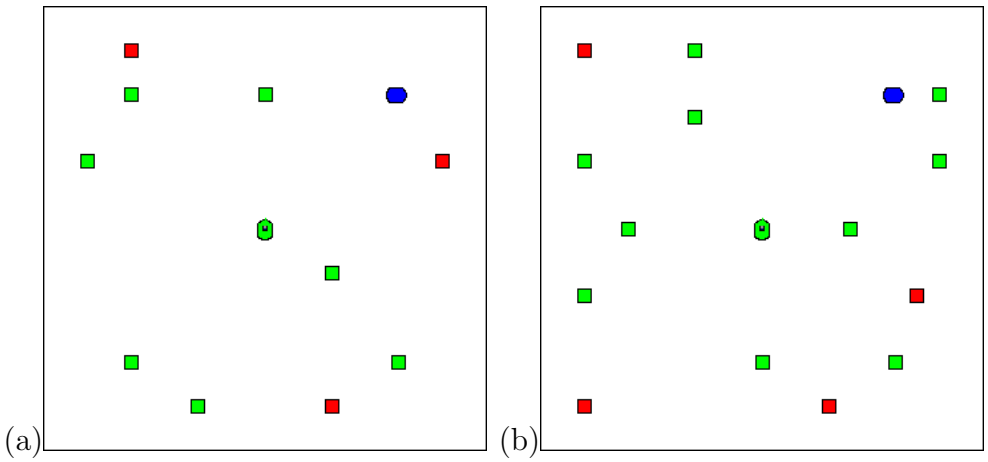


Figure 4.8: Validation Environments

possible dangerous situations = 20.8). On the other hand, we observe that such deterioration is still negligible with respect to the manual tuning case (see fitness = 0.147 and dangerous situations = 34.6).

Chapter 5

Attentive Neural Network for Real-time Applications

5.1 Introduction

In this chapter we present an implementation of the AIRM mechanisms made by using the neural network paradigm and we show the results obtained by adopting an evolutionary approach to tune some critical neuron thresholds of this AIRM-net, that regulates the overall emergent behavior of a behavior-based robotic system.

5.2 Motivations

Starting from the ARIM mechanism (Fig. 5.1-(a)) introduced in [91] we implemented the internal clock mechanism by means of a Neuro-Symbolic net (AIRM-net) [92] with the aim to face real time applications. We planned to use neural networks since they are ideally suited to the development of learning processes. These represent a powerful data modeling tool, able to capture and represent complex input/output relationships. Indeed the motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain, where the knowledge is acquired through learning directly from the data being modeled. Hence, we apply the Differential Evolution [90] technique also in this

case with the aim to find, in the space of possible solutions, the best setting of some critical parameters of the net (thresholds), regulating the overall emergent behavior. We show how this kind of algorithm is able to find the threshold values producing the best fitness and maintaining the implicit constraints introduced by the AIRM-net.

5.3 AIRM-net activity

The AIRM-net can be automatically designed by the Neuro-Symbolic Behavior Modeling Language (NSBL) [93], [94], that allows expressing propositional logical inference and translating them into the logically equivalent neural network. It is characterized by a time interval, named *clock period* (p_β), used to space out two successive sensors readings; p_β is generated by the ZEIT module (Fig. 5.1-(b)) and it is initially set to a maximum value ($p_b max$).

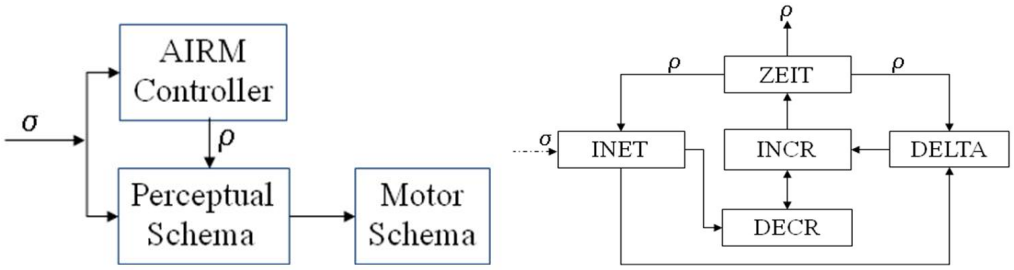


Figure 5.1: (a) Behavior schema; (b) AIRM-net controller schema.

The ZEIT module interacts with the INCR and DECR modules in order to change the clock period according to the increasing or decreasing input variations coming from INET module. Furthermore, by means of a releasing function ($InI_t or InI_\sigma$), the ZEIT module communicates when the behavior has to process sensory inputs. The AIRM-net modules are sketched in Fig. 5.2. INCR and DECR modules are controlled by INET and DELTA modules. INET conveys the input signal σ, read by the sensor at time t and $t - p_\beta$, to the DELTA and DECR modules. The DELTA module is activated when the sensor signal increases between two successive readings. The rate variation can be evaluated with respect to the

salience $\Delta\sigma_t/\sigma_t$ (where $\Delta\sigma_t = \sigma_t - \sigma_{t-p_\beta}$) or to the temporal incremental ratio $\Delta\sigma_t/p_\beta$.

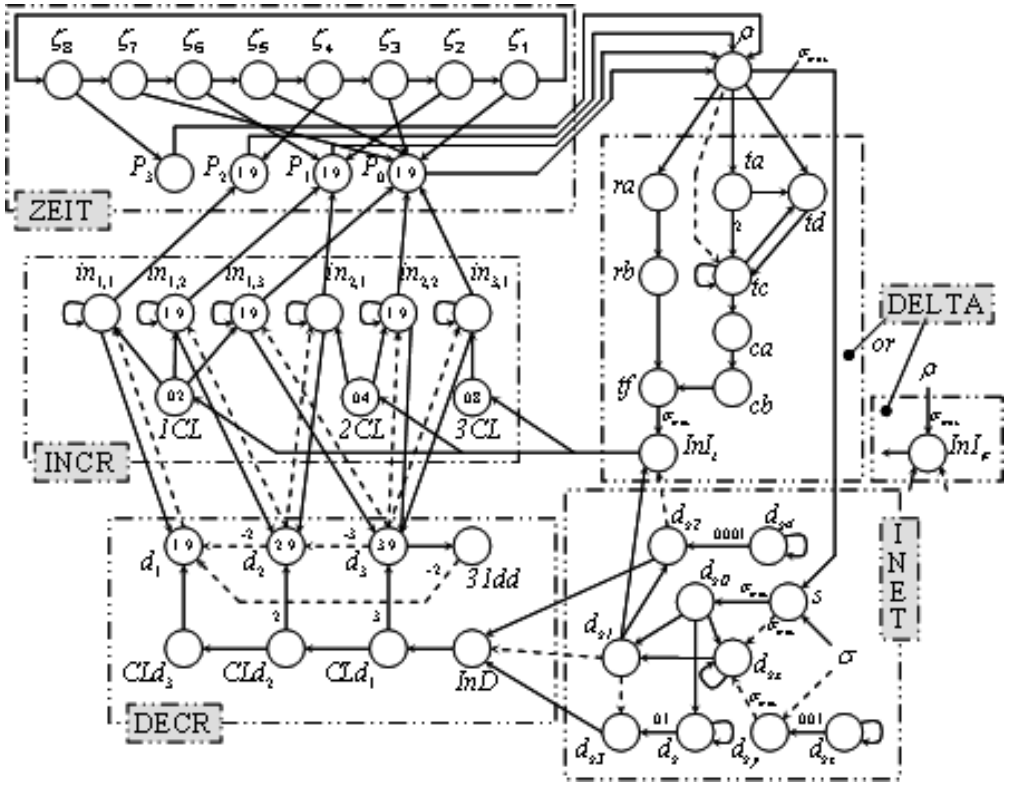


Figure 5.2: The AIRM-net.

The INET module activates the DECR module if the input signal decreases. The interaction between DECR and INCR modules and then with the ZEIT module provides respectively an increasing or decreasing of the clock period. So the INCR module (see Fig. 5.2) provides a sort of focus of attention mechanism by reducing the period p_β. This module is formed by two layers of *typeN* neurons (iCL and $in_{i,j}$) characterized by the following transfer function:

$$typeN_i(t) = 1[\sum_{j=1}^{k_i} a_{i,j} * j(t-1) - th_i]. \tag{5.1}$$

where neuron j is either a *typeN* neuron or a *type*Δ neuron. A *type*Δ neuron, such as InI_x neurons, evaluates the rate variation ($InI_\sigma = \Delta\sigma_t/\sigma_t or InI_t = \Delta\sigma_t/p_\beta$)

and fires on the *iCL* neurons of the INCR module. Without loss of generality, we chose an updating policy for the clock that decreases its period according to the powers of two. Hence, we need $n = log_2(p_{bmax})$ *iCL* neurons. If *iCL* neuron fires, the new period will be equal to $p_{bmax}/2 * i$. In a first implementation of the net we experimentally determined the values for the *iCL* thresholds, depending on several factors such as sensor precision, the special features of the environment and the behavior goal. The only constraint was that the *iCL* thresholds had to be in ascendant order in a way that the decreasing process be gradual and proportional to the variation of the input signal.

5.4 Thresholds tuning

In order to get a good performance for our robot and to extend and generalize the AIRM-net, the *iCL* neuron thresholds, regulating the period adaptation process, are tuned through an evolutionary approach; in particular, we deploy the DE algorithm. This algorithm gradually achieves the robot control system as an optimization problem. At each generation it produces a new population of candidate solutions combining the existing ones according to a mutation operator F (i.e. maintaining the most suitable solution for the optimization problem). Then, in order to increase the diversity of the perturbed parameter vectors (individuals of the new population), a crossover factor CR is introduced [90]. As we explained in previous chapter, the general formulation of the problem is to consider a *fitness function* that evaluates the system performance depending on the choice of the critical values of the *iCL* thresholds, controlling the AIRM-net, and to solve the minimization problem by finding the *iCL* thresholds combination that produces the best (minimum) value for the fitness. The fitness function evaluates the robot global behavior by considering some application-dependent performance measures (i.e. time to accomplish the goals, number of dangerous situations, etc.) during the interaction between the robot and the environment.

5.5 Test-bed Case Study

We tested our approach using a simulated Pioneer-3DX mobile robot, endowed with a blob camera and sonar sensors, and controlled by the Player/Stage tool [77].

5.5.1 Foraging Domain

The robot, without any a priori knowledge, has the task of finding food (gray circle in Fig. 5.3-(a)) in the environment, avoiding obstacles (black squares) and coming back to its nest, i.e. its starting point (striped rectangle).

This domain is a good test-bed since it combines both attractive and repulsive behaviors.

5.5.2 Attentive AIRM-Net Overview and DE approach

The behaviors are represented by suitable AIRM-net provided respectively by InI_σ or InI_t neuron. The architecture (Fig. 5.3-(b)) is characterized by three behaviors endowed by an AIRM, whose outputs are combined through the classic subsumption mechanism [7]. The DE algorithm is implemented considering as individual of a population a single robot whose AVOID AIRM-net is characterized by a particular combination of the *iCL* threshold values. The DE evaluates the performance of such a robot, while changing the *iCL* thresholds, by means of the following fitness function:

$$\begin{aligned} fitn(x) =& M_1 * \frac{avoid\_count}{cc} + M_2 * \frac{num\_crash}{cc} + M_3 * \frac{time}{task\_time} + \\ &+ M_4 * (1 - food\_count) + M_5 * (1 - nest\_reached). \end{aligned} \tag{5.2}$$

where x is an individual of the population, *cc*is the number of executed computational cycles, *avoid_count* represents the number of calls to the AVOID behavior, *task_time*is the maximum time allowed to accomplish the task, *time* is the effective time spent to accomplish the goal, *num_crash* counts how many times the robot lies beyond a prefixed distance from an obstacle, *food_found* and *nest_reached* assume

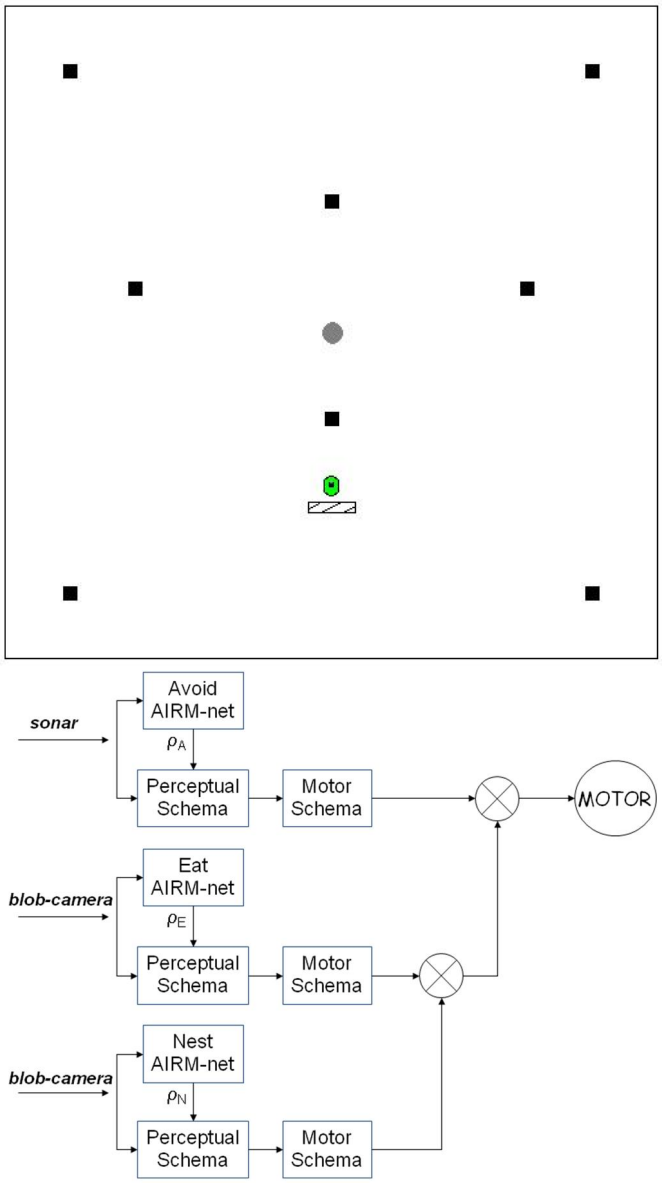


Figure 5.3: (a) Simulated environments; (b) BBR architecture.

the integer values 0 or 1 and indicate whether the robot has reached respectively the food or the nest. Also, $M_1 = 0.3$, $M_2 = 0.3$, $M_3 = 0.2$, $M_4 = 0.1$ and $M_5 = 0.1$ are constant weights and their sum must be equal to 1. These weights are chosen according to the relevance we want to assign to the parameters considered by the fitness function. Hence, fitness values will be in the range $[0, 1]$, where 1 is the

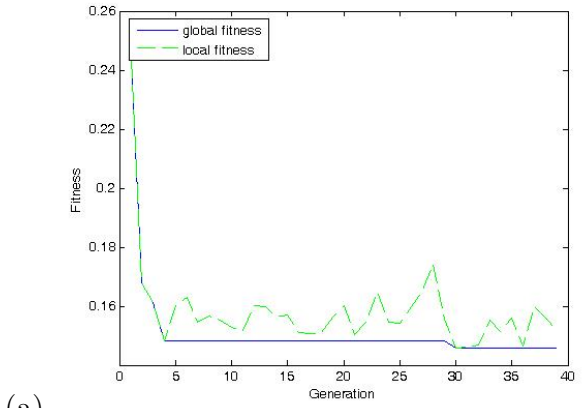
worst result and 0 is the optimum. Our goal is to tune the *iCL* thresholds in order to balance the tradeoff among these performance measures.

The DE algorithm works as already explained in Chapter 4, by initializing the starting parameters with a plausible setting. Then, it creates an initial generation G_0, of NP individuals. Each individual of the population is evaluated by means of the fitness function presented and the best element is chosen to produce the next generation. This process continues until a good fitness value is reached.

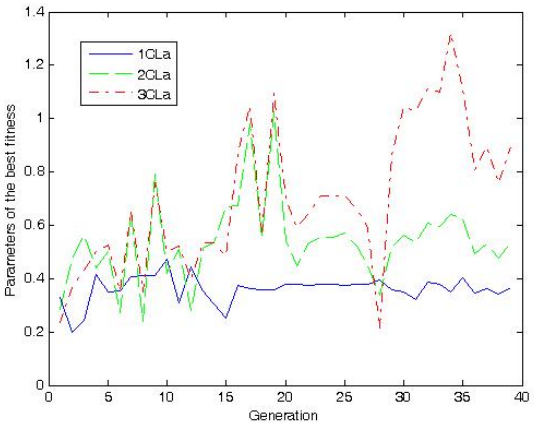
5.5.3 Experimental Results

In this section we report the results obtained by an experiment with $NP = 30$ individuals, $GEN = 37$ generations, $m = 10$ repetitions, $F = 0.85$ and $CF = 0.9$. In Fig. 5.4-(a) the evolution of the *global best fitness* (solid line) and of the *local best fitness* (dashed line) obtained at each generation is displayed. Notice that, the best fitness starts decreasing after few generations. In Fig. 5.4-(b) we also show the evolution of the AVOID AIRM-net *iCL* threshold values in the case of $p_{bmax} = 8$.

Let us highlight that, following the DE algorithm, the parameters assume values in a wide range. In the description of the AIRM-net we claimed that the thresholds of neurons *iCL* must be in ascending order. While, at the beginning, the DE algorithm randomly selects the thresholds values, at last, we find that the best fitness value is generated by *iCL* threshold values, which are again in an ascending order, coherently with the logic implied in our net. In the AVOID AIRM-net the role played by the *iCL* thresholds is to appropriately filter the $\Delta\sigma_t/p_\beta$ values provided by InI_t in order to opportunely modify the clock period. Very small values of the *iCL* thresholds imply that for small variations of $\Delta\sigma_t/p_\beta$ all the *iCL* neurons fire and then the period p_β immediately turns to 1 (similarly to the classical architectures). Very high threshold values imply less sensitivity to minor changes (more similar to an architecture with fixed periodic activations). A high fitness value is observed in both cases: in the first case because of the increase of the *avoid_count* value (see Fig. 5.4-(b) GEN=27), and in the second case due to an increase of the *num_crash* value (the sensors are checked only from time to time). In order to reduce both these performance measures, thus minimizing the



(a)



(b)

Figure 5.4: (a) Global/local best fitness; (b) *iCL* thresholds of the Avoid-net.

fitness value, we have to balance the trade off between sensitivity and periodicity by choosing uniformly distributed thresholds. In our experiment we effectively observe that a best fitness is obtained when the *iCL* threshold values are distributed in the range of values assumed by $\Delta\sigma_t/p_\beta$, once the environment has been fixed.

5.5.4 Conclusions

In this section, we proposed a neural net implementing a mechanism of periodical and adaptive activation of a robot perceptual schema, able to deal with real time applications. Moreover, in order to make this net general purpose, we employ an evolutionary approach called Differential Evolution (DE). Among different evolu-

tionary algorithms, such as Genetic Algorithms or Particle Swarm Optimization, we considered DE, since it allows, as we have noticed in the results, exploring a range of values that is not initially restricted. The results obtained by the automatic tuning of the neuron thresholds related to the AVOID behavior are very promising. Starting from these results we intend to test our model by extending the tuning also to the thresholds regulating all the behaviors of the architecture.

5.6 Conclusions

The experiments shown in this part of the thesis have been conducted for two main different purposes: (1) looking at the performance advantages obtained by using an evolutionary approach to learn adaptability; (2) understanding the role of the interaction between attentional adaptive strategies and learning made by evolutive technics. We stated that our adaptive attentional strategies in dynamic environments gain a significant advantage by the use of learning technics. We are now investigating the possibility to use both the evolutionary and standard learning techniques to improve the performance of the system. Thanks to the evolution, in fact, the robot can capture slow environmental changes, while with the standard learning technics the robot is able to detect modifications occurring during its own lifetime. The learning process might, in fact, affect the evolutionary course in an effective way since it can help and guide evolution, by adapting to changes in the environment that are too fast for the evolution to be tracked [95]. To this purpose we are currently testing the robotic architecture performance by adopting an on-line learning based on the reinforcement concept [96], [97], but the results have not yet been published.

Part IV

Deliberative Attentional System

Chapter 6

Deliberative Attentive System

6.1 AIRM for Dynamic Planning Systems

In this part we show the realization of a hybrid control architecture where the attentional mechanisms are deployed at different levels of abstraction to regulate behavioral executions, execution monitoring, and dynamic planning.

The issue of attentional processes suitable for robotic executive control has been only partially explored in the literature [98, 99]. Our aim is to provide a hybrid control architecture which integrates an attentional system [14, 100] capable of monitoring and regulating multiple concurrent behaviors at different architecture levels.

The model of attention for action proposed by Norman and Shallice assumes that a supervisory attentional system is required when a response is complex. According to this model, willed and automatic actions are controlled at different levels depending on the degree of task difficulty and complexity. For example the control operates at a lower level when the action involves an automatic response (in this case there is an increase in the activities of the cortical areas mediating selected schema); while an additional system is required when the action complexity grows. Starting from the framework presented in the previous section [68, 101], we extend this approach trying to reproduce the Shallice's supervisory attentional system in order to regulate a variety of functions, including (1) directing attention to a relevant stimulus and inhibition of irrelevant stimuli, (2) switching attention

between different parallel tasks, and (3) checking the contents of memory storage.

Within our hybrid control architecture, attentional mechanisms are used to coordinate and regulate execution monitoring and dynamic planning. We describe our system in a mobile robot case study. This domain is used as a benchmark to test the flexibility and adaptability of our system. In this context, we study the control system performance in keeping a coherent and finalized behavior while compensating failures and unexpected events.

6.2 Motivations

If we want to solve more complex problems, the introduction of a planning process, that through a reasoning system may define a long-term strategy, becomes necessary. In fact, deliberative systems, as opposed to purely reactive, behave as if they were capable of thinking, generating sequences of actions made by the combination of many elementary behaviors and especially trying to predict the effects of their actions, before executing them [102]. A deliberative module requires a strong level of abstraction, by which to provide the system with all necessary tools to operate a process of reasoning.

In fact, a government system of a deliberative robot requires, modeling almost any relevant entities within the environment; on one hand, this means translating the real world in an accurate and appropriate description, and on the other hand, determining how to symbolically represent the information on processes existing between complex real entities, in order to make the robot capable of reasoning starting from this information.

In this sense, it becomes necessary resorting to a model of the world explicitly represented in which decisions, for example, on what actions to perform, are taken through a kind of symbolic reasoning. This abstraction process requires strong assumptions on the model of the world. First, the knowledge on which to infer must be consistent and always available, and secondly it is expected that the world model used before the planning phase is still valid after its termination.

When this second hypothesis fails, indeed, there is no guarantee of achieving the desired goals, and thus a new operation planning must be performed based on the updated model of the world [103].

From a functional point of view the planner is central in constructing a deliberative robot, and can be described as a system that receives as input from the perceptual system a description of the current state and that, based on a state it wants to achieve (goal) and a set of basic knowledge, generates a plan (i.e. a sequence of actions) to execute in order to reach the desired purposes [104]. Typically a plan is constructed by analyzing the whole set of possible actions available and by choosing the best, that is the one allowing the predefined goals to be achieved as quickly and safely as possible through a specific process of reasoning. Each of these actions is represented using symbolic descriptions, and is characterized by a set of preconditions that must be met to make the plan executable, and a set of post-conditions that must be valid at its termination. The sequence of actions that constitute the plan can be obtained by searching within a data structure, usually a tree or a graph, or built from a set of rules more or less broad and from basic knowledge.

Yet, as it has been previously stated, a robotic system must be able to respond in a timely and appropriate manner to events coming from highly dynamic and unpredictable environments.

Therefore, it could be useful to constantly reschedule and update the plan to be run. This capability is called *dynamic planning*. In this way planning and execution overlap is allowed, realizing the *continuous planning* [105], so that the planner is able to formulate new goals in real time, without the performance being affected. The main issues to be addressed in this respect are the following:

- Obtain a symbolic representation of the world starting from perception, and then from the output coming from the sensors.

- Consider that the world can change after a plan has been elaborated.

- Develop plans, and possibly rework them, quickly.

We carry out this dynamic planning by adopting attentional capability to adapt the planning horizon based on the environmental circumstances and on unpredictable events.

6.3 Three-layer Architecture Endowed with AIRM

Our goal is to develop a hybrid deliberative/reactive control system endowed with attentional mechanisms which focus on sensory acquisition/processing and regulate behavior activations. The aim is to provide the executive system with a kind of supervisory attentional system [14] to suitably manage novel and stereotypical situations by combining deliberative and reactive behaviors while monitoring and regulating multiple concurrent activities [48]. More specifically, the attentional control system we propose in this thesis combines the following features:

- *Hybrid Control System.* We assume a hybrid control architecture integrating a behavior-based reactive system, an executive control system, and a deliberative system.
- *Supervisory Attentional System.* The executive control should combine reactive and goal-oriented behaviors using attentional mechanisms to orchestrate automatic reactions and activities which are scheduled on the basis of structured tasks.
- *Behavior-based Attentional System.* The overall attentional behavior should emerge from the interaction of multiple parallel attentional behaviors working at different levels of abstraction.
- *Frequency-based Attentional Monitoring.* Attentional mechanisms focus monitoring and control processes on relevant activities and external stimuli.

Our attentional control system is structured in three layers (Fig. 6.1):

- a deliberative layer that provides planning capabilities;
- an executive layer which integrates planning, execution, and plan monitoring;
- a behavior-based attentional layer that provides reactive control.

The system is designed then in a hierarchical structure divided into levels of competence, in which each layer interacts with the adjacent levels, collaborating with them and possibly exchanging data. As can be seen, it is a sort of closed

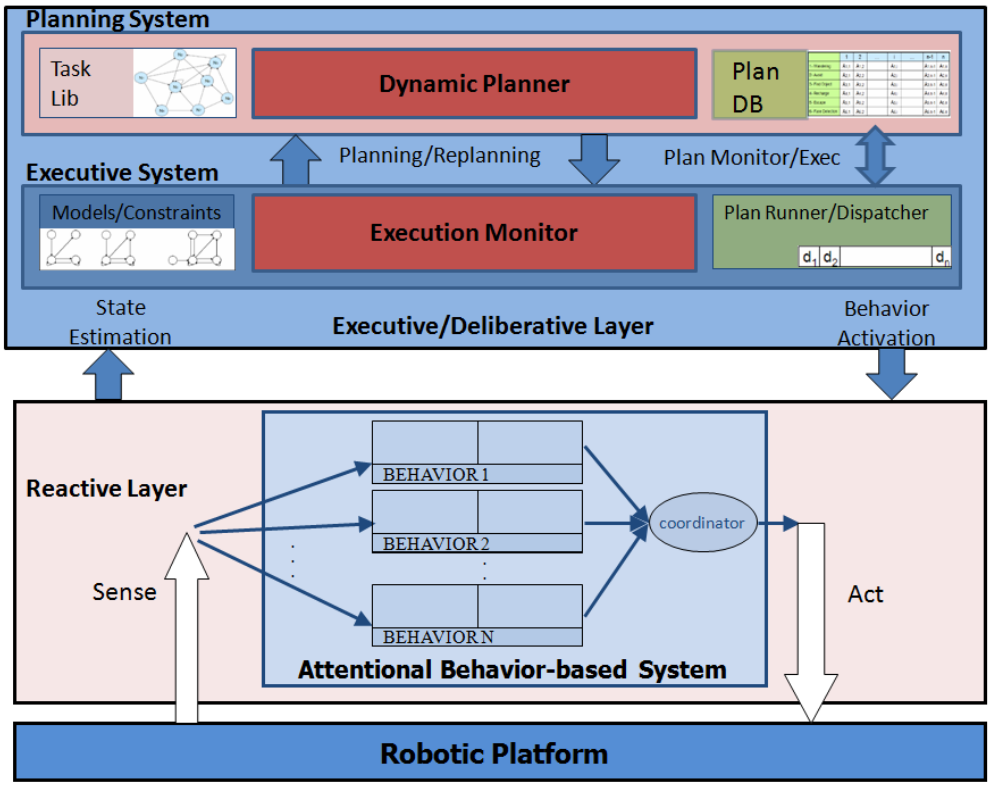


Figure 6.1: Hybrid Control Architecture.

architecture, where there is not a direct communication between the reactive layer and the planner. The architecture presented is inspired not only by the three-layer architecture [106] often proposed in the literature for the construction of hybrid systems of government, but also from the model-based configuration manager treated in [107], which defines an effective mechanism for monitoring implementation and dispatching of the instructions drawn up by the deliberative system. It has also been defined to facilitate the construction of the modules, which compone it, in a way that the system is easily extensible, but at the same time it has a sufficient set of constraints to outline the structure overall system. This modularity permits integrating highly heterogeneous components in a smooth and efficient way, also allowing changes or adds modules to the system, greatly reducing the influence of these last modules on other modules. Heterogeneous and asynchronous control structures have been used, which means that slow and fast computations can be

done in parallel, ensuring the system reacts quickly to unforeseen events. The behavior-based attentional system is of the type described before. It is composed of a set of concurrent behaviors, each endowed with an adaptive clock and an updating function. The behavior-based system produces an emergent behavior which is sufficient to control the overall robotic system when the executive/deliberative layer remains inactive. This layer provides the instinctive, automatic control along with basic motivational drives influenced by internal and external processes. The executive/deliberative system supervises the behavior executions and integrates top-down control through planning, plan monitoring, plan execution, and replanning. The executive control cycle is based on a continuous sense-plan-act control loop. For each cycle the executive system consists of: (sense) receives the current state of the system and checks for integrity constraints violations, malfunctioning, and unexpected events; (plan) calls the planner to generate or extend the agenda of future actions; (act) decides which actions are to be executed in the current execution cycle. The planning system is called at each sense-plan-act cycle with a suitable planning horizon (zero horizon for no planning). It receives the current executive and planning state and produces a plan of actions up to the end of its planning horizon. This plan (stored in the plan DB) is to be executed and monitored by the executive system.

Our architecture is thus able to integrate reactive planning, which determines the next action to take on the basis of sensory input and current state, and a long range planning term dedicated to solving complex problems (which requires consequently a higher latency time).

The attentional mechanisms are here deployed at the executive and deliberative layer in order to define adaptive monitoring and planning.

Adaptive Execution Monitoring A key role in this architecture is played by the monitor. In fact, the coexistence of traditional behavior-based architecture with a deliberative reasoning process requires a series of measures aimed at ensuring the proper conduct of their activities and proper cooperation between them (see Fig. 6.2). For this purpose an executive monitor is used to direct and coordinate the activities of the various modules involved in the execution. This entity has then the task of synchronizing the activities of coordination, to avoid conflicts

in the use of resources, enforce operational constraints and ensure safety in the presence of failures [106].

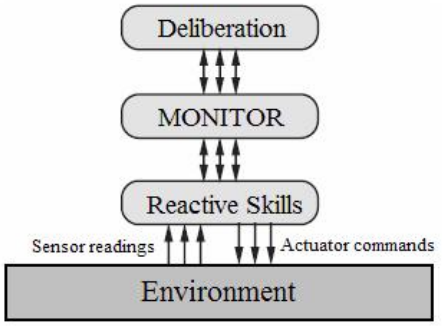


Figure 6.2: Role of the Monitor in a Control Architecture.

The task of the monitor is therefore to ensure that the system is context sensitive, by processing data of the robot and the environment in real time, monitoring the main parameters of performance and recognizing potentially dangerous situations for the robot integrity. In general, the monitoring phase can operate at low level, by obtaining information from perceptors and effectors, or at a higher level, with the task to supervise the overall robot behavior.

Analogously to the reactive layer behaviors, the execution monitor is endowed with an adaptive clock regulating the duration of the sense-plan-act cycle. The frequency of this clock depends on the attentional state of the reactive layer: the higher the frequencies of the behavioral adaptive clocks, the smaller the latency of the executive system monitoring cycle. Indeed, if the reactive processes are accelerated, the executive system should monitor and react faster reducing the duration of the sense-plan-act cycle. In our system, the approach is to regulate the executive adaptive clock period p_m^t according to the updating function:

$$f_a(\sigma(t), p_m^{t-1}) = \lambda(t, p_m^{t-1}, p_{b_1}^t, \ldots, p_{b_n}^t), \tag{6.1}$$

that is, as a function of the previous period p_m^{t-1} together with the periods of the clocks associated with the reactive attentional behaviors.

Adaptive Planning Horizon Since the executive cycle is flexible, the planning phase should be flexible too. Therefore, the planning system is also endowed with

a flexible planning horizon: a short horizon corresponds to a fast planning activity suitable for high monitoring frequencies; instead, a long planning horizon can be deployed when the system is more relaxed. Also in this case, we introduce a function that modulates the horizon length H^t at time t that depends on the previous horizon length and the attentional state of the reactive layer:

$$H^t = \Pi(t, H^{t-1}, p_m^t, p_{b_1}^t, \ldots, p_{b_n}^t). \tag{6.2}$$

For example, a simple configuration is to directly set the planning horizon H^t proportional to the latency p_m^t of the executive cycle. In this way, when the system works with no urgency (e.g. robot far from obstacles or dangerous locations) the executive system works at a low frequency that allows a more intense planning activity (e.g. plan for the next task) within a longer sense-plan-act cycle; conversely, when monitoring is intense (e.g. unexpected events or failures) only short-term planning is allowed (e.g. find quick and simple recoveries).

6.4 A Robotic Test-bed

To assess the feasibility and the effectiveness of the approach we considered a mobile robotics domain where both reactive and deliberative capabilities are needed. In our scenario, a mobile robot searches for one object requested by a human operator exploring an unknown and dynamic environment avoiding obstacles and dangerous situations while seeking sources of energy to recharge its batteries in a fixed amount of time.

The testing environment is depicted in Fig. 6.3. The robotic task can be structured into the following subtasks: search for the operator (to receive an object request); search for the requested object; go back to the operator. Each of these subtasks should be further decomposed into primitive activations or inhibitions affecting the behaviors of the reactive layer. The complete task decomposition is provided on-line by a dynamic planning system and should be continuously adapted to the executive and environmental context.

Attentional Behavior-based Layer The behavior-based attentional system is composed of the six behaviors described below (see Fig. 6.4).

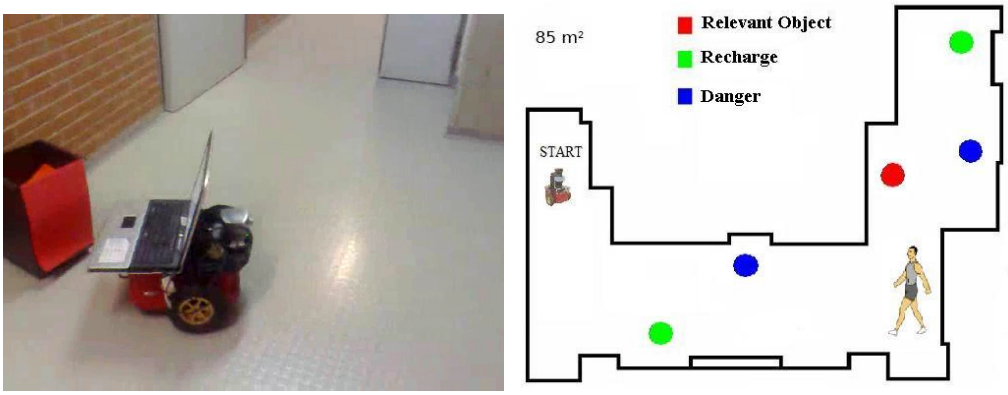


Figure 6.3: Testing Environment: the requested object is the red spot, dangerous locations are blue spots, source of energy are green spots.

`WANDER`: regulates the default random movements of the robot in the environment; we consider this as a low-level automatic behavior regulated by a constant clock period p_w^t and updating function: $f_a(void) = const_w$.

`AVOID`: provides the obstacle avoidance ability. In this case, the updating policy should allow fast reaction to dangerous potential collisions. In our setting, the `AVOID` clock period p_a^t is inversely proportional to the variation rate of the input signal (minimal distance detected by sonar sensors):

$$f_a(\sigma(t), p_a^{t-1}) = \frac{const_{a_0}}{RATE},$$

where, the $RATE$ parameter is for the first derivative of sensors signal with respect to time: $RATE = (\sigma(t) - \sigma(t - p_a^{t-1}))/p_a^{t-1}$ and $\sigma(t)$ refers to the sonar distance. Instead, when the robot is free from obstacles, it can relax by linearly increasing the period: $f_a(\sigma(t), p_a^{t-1}) = p_a^{t-1} + const_{a_1}$

`FIND_OBJECT`: provides a random search for an object in the environment (mapping/localization are not available). The frequency of this behavior increases with the time spent in the search. We introduce an updating function:

$$f_a(\sigma(t), p_f^{t-1}) = \frac{p_f max}{DESIRE},$$

where $DESIRE = const_f \times t + const_{f_0}$ represents the desire of the object which grows linearly with time according to two positive constants: as time goes by, the

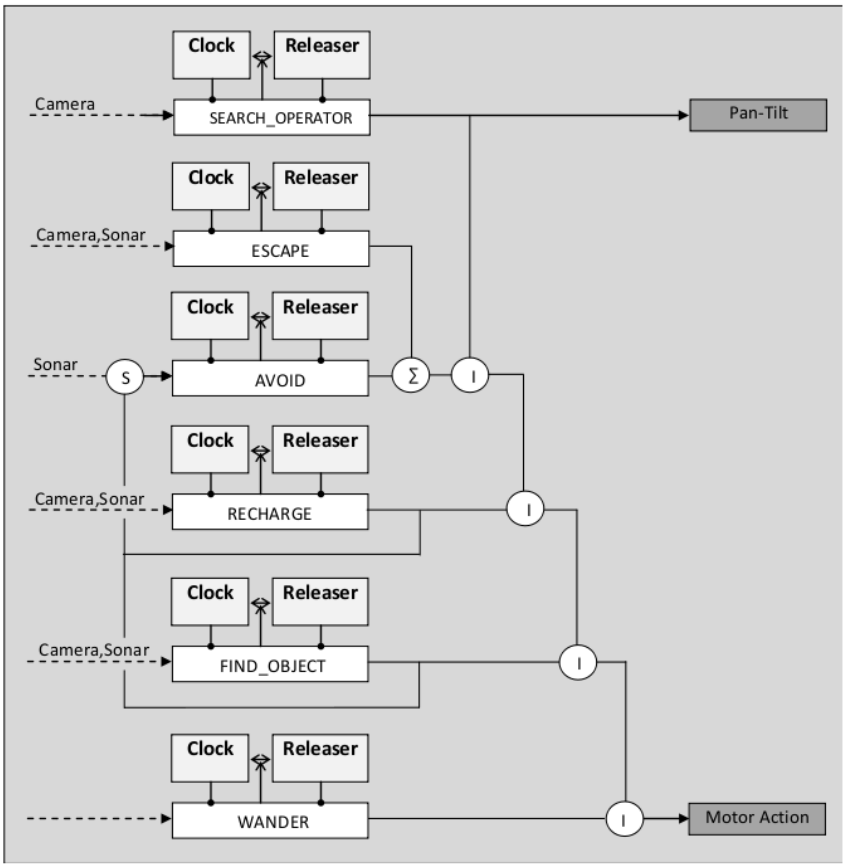


Figure 6.4: Reactive layer: behaviors and subsumption structure.

urgency of finding the target increases.

ESCAPE: determines reactive escape from dangerous situations. Its internal clock frequency depends on a possible danger in sight. Here, dangers are associated with blue blobs: if a blue blob is detected and the escape is enabled, the robot turns in the opposite direction to source of danger. We introduce the following updating function:

$$f_a(\sigma(t), p_e^{t-1}) = \frac{const_{e_0}}{FEAR},$$

where the $FEAR$ measures the degree of fear according to the Weber law of perception: $FEAR = (\sigma(t) - \sigma(t - p_e^{t-1}))/\sigma(t)$. Here, $\sigma(t)$ is the area of the perceived blue blob. This behavior is reactive and no memory is assumed. When the blue object is not perceived, the clock period is relaxed according to the linear

updating function $f_d(p_e^{t-1}) = p_e^{t-1} + const_{e_1}$.

`RECHARGE`: it allows the robot to recharge. This schema provides a random search for a source of energy in the environment. The activation frequency is here related to the charge level of the robot's battery which is modelled as a linear time-dependent updating function:

$$f_a(\sigma(t), p_r^{t-1}) = \frac{p_r max}{EN},$$

where the energy need is represented by a parameter $EN = const_r \times t + const_{r_0}$ increasing linearly with time. Below a suitable activation threshold, the lower the battery level, the greater the attention for an energy source.

`SEARCH_OPERATOR`: it looks for the face of an operator (face detection). The clock period p_{so}^t is regulated with an updating function similar to the one for $Find\_Object$:

$$f_a(\sigma(t), p_{so}^{t-1}) = \frac{p_{so} max}{DESIRE},$$

with an appropriate parameter setting.

Executive/Deliberative System The executive system monitors and regulates the reactive activities through continuous sense-plan-act cycles. The latency of this monitoring cycle is defined by an adaptive clock which is analogous to the one introduced for the reactive behaviors described above.

Adaptive Execution Monitoring Clock. In the case study, the following updating function implements (6.1):

$$f_a(t) = min(p_w^t, p_a^t, p_f^t, p_e^t, p_r^t, p_{so}^t) \times const_m. \tag{6.3}$$

Thus, the execution monitor attentional state (i.e. sense-plan-act cycle duration) depends on the most excited reactive behavior.

Executive Model. At the executive layer, each reactive behavior is explicitly represented by a finite state automaton while the allowed concurrent state transitions are limited by sets of constraints.

In Fig. 6.4 we show six automata used to represent the executive state of the reactive behaviors. From the monitor point of view, each behavior can be in an *active* state, when activated by the releaser or by the executive, otherwise, it can be *idle*, i.e. waiting for activation, or *inhibited* by a reactive subsumption mechanisms

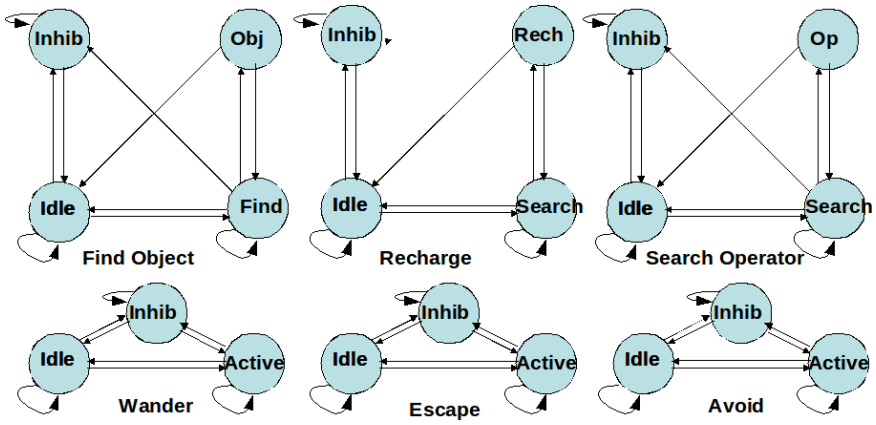


Figure 6.5: Executive Models: each behavior is associated with an automaton

or by the executive itself. For the goal-oriented behaviors like $Find\_Operator$, $Find\_Object$, and $Recharge$, we also introduced a goal-state (e.g $Operator\_Found$ for $Search\_Operator$). Given these six automata, the overall executive state s_m is described by the tuple $(s_1, s_2, \ldots, s_6)$ collecting the current executive state for each behavior. Furthermore, the automata transitions $\delta(s_i, a, s_j)$ (from state s_i to s_j with a either a control action or an external event) are synchronized and limited by a collection of constraints γ. We represented γ as a tuple (S^+, S^-, R, T) where S^+ and S^- are, respectively, set of states enabling and disabling the transition $\delta(s_i, a, s_j)$, analogously R and T are sets of resource and time constraints. In particular, we introduced the following constraints. $Escape$, $Wander$, and $Recharge$ are mutually exclusive (i.e. cannot be active at the same time), the same holds for $Find\_Object$ and $Find\_Operator$, which are also incompatible with $Escape$ and $Recharge$. As for resource constraints, we consider two battery consumption constraints: $lowBattery$ enables the activation of $Recharge$; $outofBattery$ imposes the transition to *inhibited* for each behavior; while $not\,outofBattery$ is the precondition for any activation transition. As for time constraints, we considered only two *time_out* constraints imposing a maximal duration (1/4 of the mission time) for $Face\,Detection$ and $Find\,Object$.

Execution Monitoring and Planning During the sense phase, the executive system estimates the current executive state $(s_1, s_2, \ldots, s_6)$ and checks for con-

straint violations and plan consistency (if a plan is available). If the current state is safe and the executed transitions are coherent with respect to the dispatched ones, then the execution monitor calls the planner to extend the current sequence of planned actions; otherwise, all the future activities are to be replanned from the current state. Once the plan is ready, the dispatcher selects a set of commands to be executed in the execution cycle.

Hierarchical Task. Our planning system searches the plan within the hierarchical (and incomplete) task network that partially specifies the robot activities. In our case study, the main robotic task is defined as follows:

(1) **main**(*loop*(`find_op` ; `find_obj` ; `bring_op`)),
(2) **find_op**(*search*(`wander`) ; *search*(`op_found`)),
(3) **find_obj**(*search*(`obj_found`)),
(4) **bring_op**(*search*(`op_found`)).

Here, $search(x)$ stands for: search any executable sequence of transitions yielding to the state x. Therefore, (2) is for any sequence which permits the activation of *wander* and ends with the operator found; (3) stands for any sequence of commands ending in object found; (4) is for any sequence ending with the operator found. (1) describes the main control loop: find an operator; find the object; go back to the operator.

Planning Task. Given the current initial state $(s_1, \ldots, s_6)$, an incomplete plan $plan_I$ and a planning horizon H, our planner searches from $(s_1, \ldots, s_6)$ an executable sequence of transitions (i.e. enabled by the constraints) in the automata $A_1, \ldots, A_6$, of maximal length H, which completes $plan_I$.

Planning Algorithm. Given a cost function $c(s, a, s')$ associated with the automata transitions, we deploy an A^* algorithm where the heuristic function he exploits the hierarchical task structure. Namely, given the executed transitions $\alpha = a_1; \ldots; a_n$, the current state $\sigma = (s_1, \ldots, s_6)$ and the remaining plan $plan_r$, the total estimated cost is given by $c(\alpha) + he(\sigma, plan_r)$, where $c(\alpha)$ is the total cost of the executed actions and $he(\sigma, plan_r)$ estimates the cost of executing $plan_r$ from σ. In our case, $he(\sigma, plan_r)$ is an ad-hoc function that underestimates the plan execution cost. Roughly, given the minimal action path $min(plan_r, \sigma)$ that executes $plan_r$ from σ neglecting constraints, we state $he = min(plan_r, \sigma)$.

Adaptive Horizon. The adaptive horizon (6.2) is implemented as follows:

$$H^t = \frac{maxHorizon \times p_m^t}{p_m max}, \tag{6.4}$$

that is, the current horizon length H^t is a fraction of the $maxHorizon$ proportional to the fraction of p_m^t, length of the execution cycle, with respect to the maximal cycle length $p_m max$.

Plan Execution. The generated plan is stored in the plan-DB structure which is the agenda of the future commands (events) to be executed (observed) by (from) the reactive system (Fig. 6.4).

				Current Action ⇩ i			Last Planned Action ⇩ n
	1	2	...		...	n-1	
WANDER	Active	Active		Active		Idle	Active
AVOID	Active	Active		Ignore		Inhibit	Active
FIND_OBJECT	Search	Ignore		Ignore		Inhibit	Search
RECHARGE	Ignore	Ignore		Search		Inhibit	Ignore
ESCAPE	Idle	Active		Ignore		Idle	Ignore
SEARCH_OPERATOR	Idle	SearchFace		Idle		SearchFace	Idle

Figure 6.6: Generated plan and executed activities in the plan-DB.

Given the current state, the dispatcher selects a set of commands scheduled in the plan-DB to be executed in the next step. The executive system can either force the activation/inhibition of the behaviors or just wait for their spontaneous evolution. From the reactive system perspective, the executive commands are seen as additional (top-down) releasing/inhibition mechanisms. Once a set of commands are sent to the executive system, the latter waits for the action executions till the end of the execution cycle latency p_m^t, then the executive cycle can restart from the sense phase.

6.5 Empirical Results

To assess the effectiveness of attentional monitoring and planning, we measured the system performance in the presence of external events disturbing the execution. More specifically, we forced randomized activations of escape at different

frequencies (25, 15, or 5 seconds). For each test, the configuration is the one in Fig. 6.3. The obtained results are compared with respect to the ones gathered with fixed regulations for the monitoring clock and the planning horizon. In this scenario, we tested: (a) at the reactive level, the efficiency of the attentional reactive system in terms of number of activations needed to achieve the task; (b) at the executive level, the executive system effectiveness in keeping a coherent executive behavior despite the disturbances; (c) at the deliberative level, the performance of the adaptive planner in keeping a finalized behavior despite the environmental disturbances. For each test case, we collected average and standard deviation of 15 runs, with 300 seconds as maximum duration of each run.

Platform. As a robotic platform we used a Pioneer 3DX endowed with a blob-camera. The control system runs on an Acer 9504WSMi, Intel Pentium M 760 2.0 Ghz, 1 Gb RAM DDR2, S.O. Ubuntu 9.11.

Attentional Behavior-based System. In the first place, the attentional system should reduce the overall sensory readings and behaviors' activations needed to achieve the task. This effect is evident in Tab. 6.1 where we show the performance (mean and variance of sensory readings per minutes) of our system (Adaptive Clock column) compared with respect to the one of the same system endowed with a clock period fixed at 100 ms (Fixed Clock column). These results are collected assuming disturbing events generated at about 15 seconds and the planning activity disabled (planning horizon reduced to zero). In all the test cases, the agent could accomplish the task within the allowed time.

Table 6.1: Sensory Readings (readings/min)

Behavior	Adaptive Clock	Fixed Clock
Avoid	228 ± 51	492 ± 104
Find-Object	73 ± 25	219 ± 69
Escape	113 ± 29	292 ± 94
Recharge	92 ± 28	225 ± 64
Search-Operator	54 ± 17	198 ± 39

Attentional Execution Monitoring. To test the adaptive monitoring, we measured the detected failures with respect to the execution monitoring latency. In

particular, the executive system incurs in two kinds of failures: (1) constraints violations, i.e. the reactive system violates some γ constraints defined in the executive model, and (2) execution failures, i.e. the executed actions/events differ from the expected ones. Note that these failures are physiological in our system. Indeed, the reactive system has its internal drives and can easily become misaligned with respect to the executive expected state; however, a good regulation of the monitoring period should reduce this effect. Also in this case, we assumed noisy events generated at about 15 seconds. The planning horizon is here short and fixed (we considered a horizon of 2 steps ahead) to assess the adaptive executive without the adaptive planning effect, while reactive clocks are adaptive. The results are reported in Tab. 6.2. Here, for higher monitoring frequencies we obtain more failures. This is due to the combined effect of misalignments (expected transitions not executed within a cycle) and false positive constraint violations. Instead, the adaptive regulation seems the best setting.

Table 6.2: Constraints violations and execution failures

Clock (ms)	Const. Violations (%)	Exec. Failures (%)
100	2.7 ± 2.3	4.0 ± 2.3
200	2.4 ± 1.6	4.0 ± 2.0
400	1.5 ± 1.3	2.2 ± 1.7
800	1.1 ± 0.9	1.9 ± 1.3
1200	1.0 ± 0.9	2.3 ± 1.7
Adaptive	1.0 ± 0.9	1.7 ± 1.1

Attentional Planning/replanning. To test the adaptive planning effectiveness, we compared the robot performance with fixed and adaptive horizons (with reactive and executive clocks adaptive in both cases). The performance is measured as the average time to accomplish the mission. We first considered the fixed horizon case in two settings with noisy events generated at 25 and 5 seconds respectively (see Fig. 6.7). Interestingly, while in the first environment, as expected, the longer the horizon the better the performance, in the second one we register an anomaly: after a certain length, long horizons start to become counter productive. This negative effect is due to disturbances and continuous replanning that degrades the task-based control coherence.

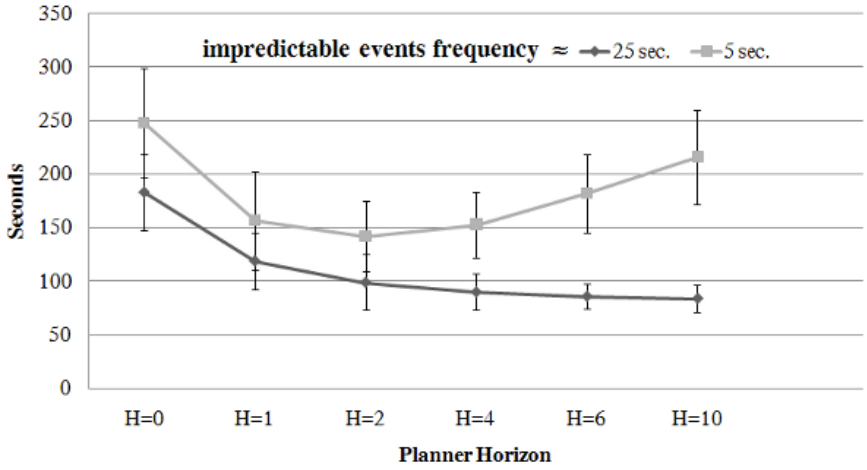


Figure 6.7: Fixed Horizon.

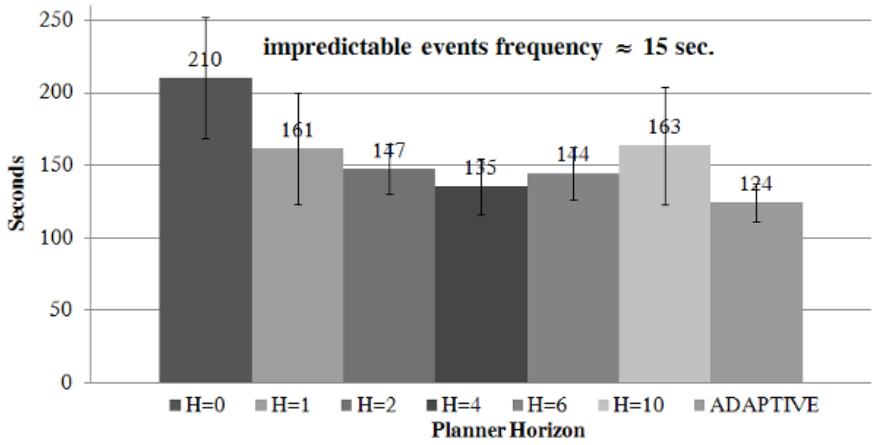


Figure 6.8: Adaptive Horizon.

In Fig. 6.8, we consider a mixed scenario with disturbing events (randomly generated from 5 to 25 s with mean in 15 s.) comparing the performance of fixed and adaptive horizon regulations. Here, the adaptive horizon seems to perform better than any fixed regulation.

6.6 Conclusions

We have presented an attentional hybrid control architecture where simple attentional mechanisms are used at different level of abstraction. We exploited a frequency-based model of executive attention to regulate and coordinate reactive behaviors, execution monitoring, and dynamic planning. The executive system supervises the behavior executions and integrates top-down control adapting its sense-plan-act latency to the behavioral activation/excitation level. Moreover, the sense-plan-act cycle duration regulates the length of the planning horizon. This allows us to adapt deliberation and reactivity to the attentional state of the system through the execution. We implemented the proposed architecture in a robotic case study testing its performance under different conditions. The collected results illustrate the advantages of the adaptive regulation with respect to other settings with fixed horizons and monitoring latencies. In general the proposed architecture allows combining the advantages of both the purely reactive and deliberative systems. The robot exhibits a behavior characterized by a sufficient degree of complexity while retaining the ability to extricate themselves successfully in situations that require on one hand a response in real time, on the other hand the ability to deal with unforeseeable events.

Part V

Application of the AIRM to HRI Tasks

Chapter 7

Human-Robot Interaction guided by Attention

7.1 Introduction

In this section we apply our attentional mechanism to a human-robot interaction task with the aim of balancing the trade off between safe human-robot interaction and effective task execution. These mechanisms allow the robot to increase or decrease the degree of attention toward relevant activities modulating the frequency of the monitoring rate and the speed associated to the robot movements. In this framework, we consider pick-and-place and give-and-receive attentional behaviors. To assess the system performance, we introduce suitable evaluation criteria taking into account safety, reliability, efficiency, and effectiveness.

7.2 Motivations

A robotic system designed to physically interact with humans should adapt its behavior to the human actions and the environmental changes in order to provide a safe, natural, and effective cooperation. The human motions and the external environment should be continuously monitored by the robotic system searching for interaction opportunities while avoiding dangerous and unsafe situations. In this context, we propose to use our attentional system for balancing the trade off

between safe human-robot interaction and effective task execution.

Human aware manipulation [108, 109] and human-robot cooperation in manipulation tasks [110, 111] are very relevant topics in HRI literature, however cognitive control and attentional mechanisms suitable for safe and effective interactive manipulation are less explored. A number of recent contributions about close HRI are based on motivational and cognitive models [112]. However, attentional mechanisms in HRI have been mainly investigated focusing on visual and joint attention [113, 112] for social interaction. In contrast, our main concern is on (supervisory) executive attention for orchestrating the human-robot interaction activities monitoring their safety and effectiveness [14, 100].

We assume the presented frequency-based model of the executive attention [58, 87] where each behavior is endowed with an adaptive internal clock that regulates the sensing rate and action activations. In our human-robot interaction domain, the attentional mechanisms regulate two conflicting requirements: (1) safe interaction with the humans; (2) effective cooperation in interactive tasks. Depending on the disposition and the attitude of a person in the environment, the sensing rates and behaviors activation frequencies are increased and decreased changing the overall attentional state of the system. For example, a person approaching the robot workspace or an abrupt movement of his/her hands affects the attentional process of the robot that determines a more frequent elaboration of this perceptual input (human movements), with respect to other inputs or the execution of other tasks, and a slower movement of the robotic manipulator.

In particular, as a case study we consider simple robot manipulation tasks providing the attentional monitoring strategies for behaviors like pick and place, give and receive, search and track (humans and salient objects). To assess the attentional system performance, we introduce suitable evaluation criteria (safety, reliability, efficiency, and effectiveness). The empirical evaluation shows the advantages of attentional system with respect to non-attentional versions of the same framework (non-adaptive clocks).

7.3 Attentional HRI Model

Our aim is to develop an autonomous robotic system suitable for human-robot interaction in cooperative manipulation tasks. Achieving autonomy and safety in such a environment requires adaptation. For this purpose, we propose to deploy our AIRM attentional system to modulate the robotic arm motion and perception in order to achieve an effective coordination and interaction with the human movements in the operative space. Here the attentional mechanisms regulate the executive system trading of two conflicting requirements:

- (i) safe interaction with the humans;
- (ii) effective cooperation in interactive tasks.

Each requirement is associated with a motivational drive that affects the attentional and executive state of the robotic behavior. The first one corresponds to the fear of hurting people, and thus it determines caution, slow movements and intensive monitoring (in case of danger it blocks the robot motion); instead, the second one is associated with a desire to interact with people and manipulate objects, and thus this attitude provides an attraction towards moving and close persons or objects.

Depending on the disposition, movements, and the attitude of a person in the workspace, each behavior changes its activation frequency, affecting the overall attentional state of the system. In this way, a person walking across the interaction area or a fast movement of a human head (or hand) can modify the attentional state of the same behaviors causing an accelerated elaboration of the associated perceptual input (human movements) and a more frequent behavior activation.

7.4 Test-bed Case study

7.4.1 Manipulation domain

We considered a robotic manipulator (see Fig. 7.1) that is to cooperate with a human operator in pick-and-place and give-and-receive (hand-over) tasks. Depending on the context, the robotic system should: look for an operator to interact with;

give or receive an object to/from the operator; pick or place an object from/into a location. Each of these tasks is to be monitored in order to avoid dangerous/unsafe situations.

In this context, the attentional mechanisms allow us to combine the robot attraction towards the operator and the robot repulsion from unexpected events and abrupt environmental changes. For each behavior, the simple perception-action response to an external stimulus may produce different patterns of interactions depending on different internal states of the robot given by the combination of the fear of hurting the user and the desire of helping him/her.
Platform. We used a PIONEER 3DX endowed a 7DOF robotic arm (Cyton Arm by Energid: payload 300 g, hight 60 cm, reach 48 cm, joint speed 60 rpm), a gripper (size 3.25 cm) as end-effector, and a stereo-camera (Videre Design LLC, baseline 9 cm, 640x480, 64 disparity, 30 Hz) for visual servoing (see Fig. 7.1). The robot is controlled by a Player/Stage client [77]. All the robot behaviors are implemented in a cycle using a single thread of execution.

Environment. In our setting, the robot base is kept fixed (the mobile base is not exploited) and close to a small table where the robot can pick and place objects. Depending on the proximity, we defined three areas in the workspace: a proximity area (10 cm from the robot body, too close for safe HRI); an interaction area (10–50 cm, where physical human-robot interaction is possible. Here we refer to both visual and physical interaction in the robotic arm workspace); a far workspace area (from 50 cm to 6 m, humans and object in the robot field of view, but too far for objects hand-over).

7.4.2 Control Architecture Overview

We designed a control architecture suitable for the primitive interactive manipulation tasks introduced above. The control system integrates modules for direct, inverse kinematics, and visual servoing along with modules for face recognition, hand detection/tracking, object recognition/tracking. In particular, inverse kinematics and visual servoing are based on a CCD (Cyclic Coordinate Descent) algorithm [114] suitable for fast and continuous adaptation of the robot arm motion with

Figure 7.1: Robotic platform (left), workspace (right).

respect to the environmental changes. Face detection is based on Viola and Jones algorithm [115], while hand and object detection/tracking are based on simple skin and blob detection algorithms. Given these functionalities, the attentional state of the robot is affected by the following sources of saliency: face, hands, objects detection, proximity.

Attentional Behaviors The behavior-based architecture is depicted in Fig. 7.2. This model integrates attentional behaviors for pick and place, give and receive, but also behaviors for search and track (humans and objects) as well as behaviors regulating the avoid attitude of the robotic system.

The robot attentional behavior is obtained as the combination of the following primitive behaviors (see Fig. 7.2): `AVOID`, `PICK` and `PLACE`, `GIVE` and `RECEIVE`, `SEARCH` and `TRACK`. For each behavior, we have to define the activation function and the updating policy that represents the attentional model associated with.

`SEARCH` controls the pan-tilt (PTU) providing an attentional scan of the environment looking for humans and objects. It is active whenever the robotic system

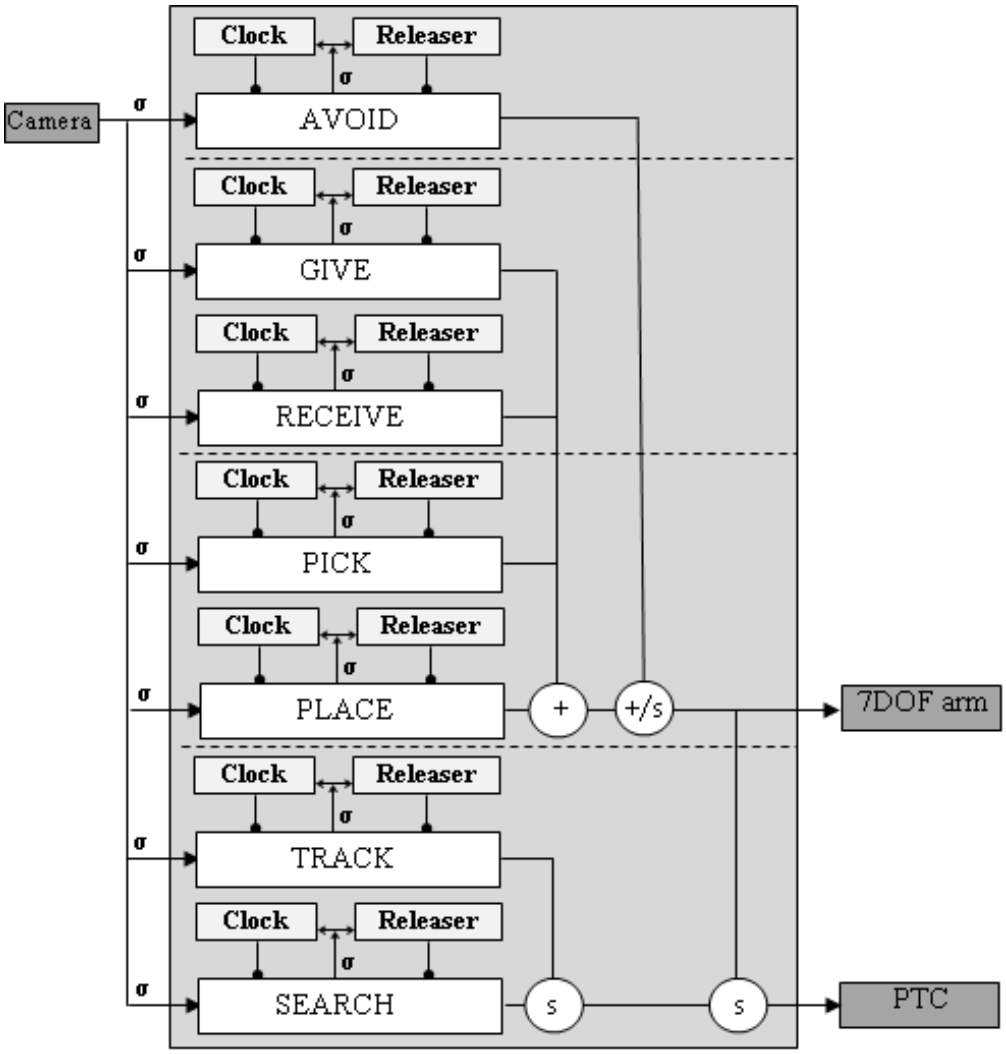


Figure 7.2: Behavior-based architecture for HRI.

is idle and no interesting things (objects or humans) are in the robot field of view. Its activation is periodic, but not adaptive, hence it is associated with a constant clock:

$$p_{sr}^{t} = const_{sr}.$$

Once a human is detected in the robot workspace (through face detection and/or hand detection), the `TRACK` behavior is enabled. This behavior allows the robot to monitor humans motions before they enter in the interaction space. `TRACK`

focuses the system attention on the operator movements, hence the adaptive clock should be regulated in accordance with the human motion and position. Here, the input signal $\sigma_{hm}(t)$ represents the human distance from the robot camera; in our test-bed, it is the minimal distance of human faces and hands. The `TRACK` clock period changes according to $\sigma_{hm}(t)$ and the increment of $\sigma_{hm}(t)$, that is, the period p_{tr} is updated as follows:

$$p_{tr}^{t} = \Theta_{tr}(\sigma_{hm}(t), \frac{\sigma_{hm}(t) - \sigma_{hm}(t - p_{tr}^{t-1})}{p_{tr}^{t-1}}),$$

where p_{tr}^{t-1} is the period at the previous clock cycle, $\Theta_{tr}(x, y)$ is a function $\Theta_{tr}(x, y) = \phi_{tr}(\alpha x + (1 - \alpha)1/y + \beta)$, where α and β are behavior specific parameters used to weigh the importance of position and velocity in the attentional model, while $\phi_{tr}(z)$ is the scaling function that introduces suitable thresholds to keep the clock period within the allowed interval $[p_{tr}min, p_{tr}max]$. Intuitively, a human that moves fast and close needs to be carefully monitored (high frequency, foreground), while a human that moves far and slow can be monitored in a more relaxed manner (low frequency, background).

The `AVOID` behavior checks for safety in human-robot interaction, it controls the arm motion speed and can stop the motion whenever a situation is assessed as dangerous. `AVOID` is enabled when a human is detected in the robot interaction area. It is endowed with an internal clock whose frequency depends on the operator proximity and motion. The associated clock frequency changes proportionally to the situation saliency. That is, if the operator is close and/or its position σ_{op} (i.e. minimal distance of face and hands) becomes closer between successive readings of sensory data, then the clock is accelerated, while it is decelerated if the operator moves away from the robot. The period of this clock changes as follows:

$$p_{av}^{t} = \Theta_{av}(\sigma_{op}, \frac{\sigma_{op}(t) - \sigma_{op}(t - p_{av}^{t-1})}{p_{av}^{t-1}}),$$

where Θ_{av} is defined as for `TRACK`. The output of this behavior results in a speed

deceleration associated with high frequencies:

$$speed = \begin{cases} \frac{max\_speed \times p^t_{av}}{p_{av} max} & \text{if } prox.sp. < \sigma_{op} \leq int.sp. \\ 0 & \sigma_{op} \leq prox.sp. \end{cases}$$

where *speed* is the current speed, *max_speed* is the maximum allowed value for the arm speed. Moreover, the arm will stop if the operator is inside the robot space (proximity space).

The `PICK` behavior is activated when the robot is not holding and object, but there exists a reachable object in the robot interactive space. `PICK` moves the robot end-effector towards the object, activates a grasping procedure and, once the robot holds the object, moves this in a predefined safe position close to the robot body. For `PICK`, the input signal $\sigma_{obj}(t)$ represents the distance of the object from the robot end-effector which can be detected by the stereo-camera. In this case the clock period is associated with the distance of the object. That is, the period p^t_{pk} is updated as follows:

$$p^t_{pk} = \phi_{pk}(\alpha\, \sigma_{obj}(t)), \tag{7.1}$$

where $\phi_{pk}(x)$ is the scaling function used to scale and map $\sigma_{obj}(t)$ in the allowed range of periods $[p_{pk}min, p_{pk}max]$. Furthermore, the clock frequency determines also speed variations. In particular, the speed is related to the period according to the following relation:

$$speed = \frac{max\_speed \times p^t_{pk}}{p_{pk} max}. \tag{7.2}$$

In this way, the arm moves with *max_speed* at the beginning, when there is free space for movements (and a low monitoring frequency), and smoothly reduced its speed to a minimum value in order to execute a precision grip with more frequent camera information (higher monitoring frequency).

As for `PLACE`, it is activated when the robot is holding an object in the absence of interacting humans in the interactive space. It moves the robot end effector towards a target position, it places the object and moves the robot arm back to a predefined position close to the robot body. The clock period is regulated

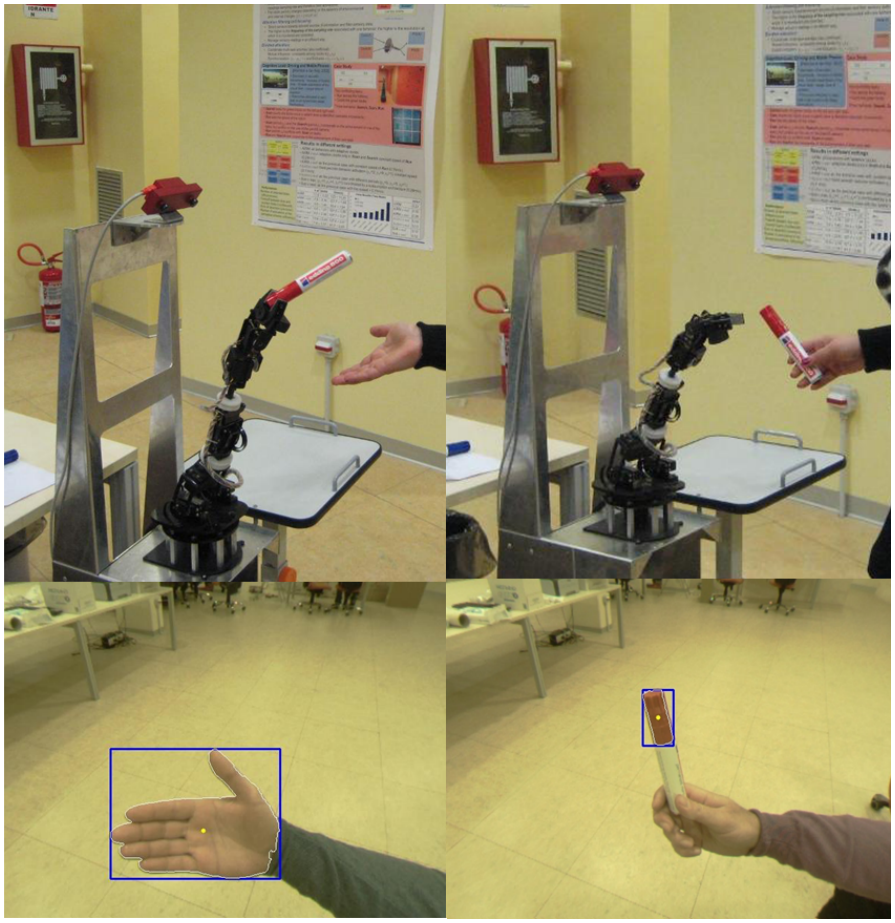


Figure 7.3: Human-robot interaction schemas.

by a function analogous to the (7.1) with the distance to the target σ_{tr} as the input signal. Also in this case, the speed is decelerated at high clock frequencies according to (7.2).

The `GIVE` and `RECEIVE` behaviors are activated by object and gesture detection. These behaviors are responsible for monitoring and regulating the activities of giving and receiving objects taking into account both the humans proximity and their movements. In this case, the clock period is associated with the distance of both the objects and the speed of the operator hand. In particular, `GIVE` is activated when the robot holds an object and perceives a reachable human hand in its operative space. When activated, this behavior moves the end-effector in

the direction of the operator hand with a trajectory and velocity which depends on the human proximity and operator hand movements. The `GIVE` sampling rate is regulated by the following function:

$$p_{gv}^{t} = \Theta_{gv} \begin{pmatrix} \gamma_{obj}(\|\sigma_{obj}(t) - ee_{pos}(t)\|), \\ \gamma_{op}(\frac{\sigma_{op}(t) - \sigma_{op}(t - p_{gv}^{t-1})}{p_{gv}^{t-1}}) \end{pmatrix}, \tag{7.3}$$

where $\sigma_{obj}(t)$ and $ee_{pos}(t)$ are the positions of the object and the end-effector at time t, $\sigma_{op}(t)$ is the hand operator position, θ_{gv}, γ_{obj}, γ_{op} are suitable functions defined as follows. The function γ_{obj} sets the period proportional to the object position, i.e., the closer the object, the higher the sampling frequency:

$$\gamma_{obj} = (p_{gv}max - p_{gv}min)\frac{d}{max_d} + p_{gv}min,$$

where d, max_d are, respectively, the distance $(\sigma_{obj}(t) - ee_{pos}(t))$ and the maximal distance between the end-effector and the object. Instead, γ_{op} depends on the hand speed v (in terms of the incremental ration of the hand position towards the value of the period), i.e., the higher the speed, the higher the sampling frequency. The following function is used to set and normalize the values within the allowed interval:

$$\gamma_{op} = \begin{cases} (p_{gv}max - p_{gv}min)(1 - v) + p_{gv}min & \text{if } v \leq 1 \\ p_{gv}min & \text{otherwise} \end{cases}$$

Finally, $\Theta_{gv}(x)$ combines the two functions γ with a weighted sum regulated by an α parameter

$$\Theta_{gv}(x) = \phi_{gv}(\alpha\gamma_{obj} + (1 - \alpha)\gamma_{op})),$$

also in this case the resulting period is limited within the allowed interval $[p_{gv}min, p_{gv}max]$ by the scaling function ϕ_{gv}.

The clock frequency regulates not only the sampling rate, but also the velocity of the arm movements. More specifically, the execution speed is related to the period according to an inversely proportional relation according to (7.2). This implies that the higher the sampling rate, hence the attention, the slower the hand movement. Intuitively, here we assume that when attention is needed the

movement should be more carefully monitored, and thus slowed.

As for the RECEIVE behavior, it is activated when the robot perceives a human in the operative space holding a reachable a object in his/her hand. The behavior sampling rate is regulated by a function analogous to (7.3) (set with different parameters) with an adaptive velocity inversely proportional to the current period, as in (7.2).

7.4.3 Execution Example

We now illustrate how the system works in typical interactive situations. In Fig. 7.4 we plotted part of the execution of the RECEIVE behavior. In particular, Fig. 7.4-(a)

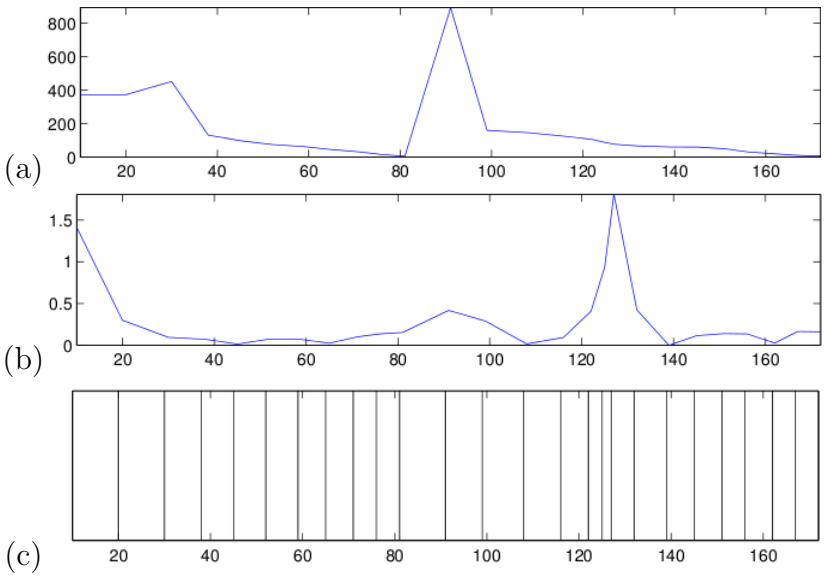


Figure 7.4: (a) End effector-hand distance; (b) Hand speed as evaluated by the Receive Behavior; (c) Activations of the Receive behavior.

represents the variation of the distance between the end effector of the robotic arm and the operator hand. In the execution Cycle 80, the robot has almost reached the human hand, however the operator moves his/her arm away. The execution of the behavior ends at the execution Cycle 162 when the robot delivers the object to the operator. Figure 7.4-(b) represents the hand speed variation of the same execution, as evaluated by the RECEIVE behavior. The hand is almost stationary

between Cycle 30 and Cycle 70, then it starts moving with different speeds until it stands still at Cycle 162 and receives the object. Finally, Figure 7.4-(c) represents the activations of the behavior at each cycle. Whenever there is a bar in the plot, this means that the behavior perceptual schema is active. Let us note that both the distance and the hand speed are sampled and evaluated only when the behavior perceptual schema is active. The frequency of activation will increase when the distance is small (for example between Cycles 40 and 80) or when the hand speed is high (for example between Cycles 105 and 125) following the updating function of the behavior.

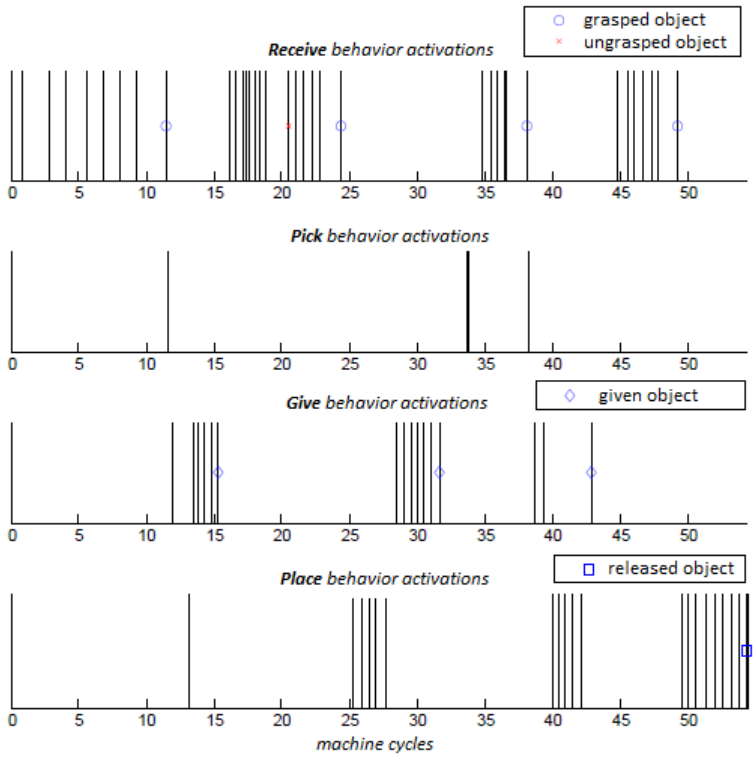


Figure 7.5: This figure shows the activation of all the attentional behaviors: receive, pick, give and place. Each vertical line corresponds to an activation of the corresponding behavior.

In Fig. 7.5, we show a complete run of the system and how the different attentional behaviors are activated during human-robot interaction schemas. Initially, the user is holding an object; this fact leads the behavior *Receive* to rise up its

activation frequency until the arm successfully grabs the object at the execution Cycles 11. Then, the robot waits keeping both *Place* and *Give* behavior active until the operator shows his/her hand in the robot arm workspace. In this case in the third row of Fig. 7.5 we can see that the behavior *Give* is activated more frequently than the *Place* behavior, until Cycle 16 in which the object is released. Now the user has again the object in his hand, therefore the *Receive* behavior is activated more frequently until the Cycle 21 in which the robot tries to grasp the object, but without success (red x on the graph). The robot can execute another attempt and reaches the object at the Cycle 24. The operator is now not interested in receiving the object and his hand is out from the robotic arm workspace, so the *Place* behavior is activated more frequently until, at Cycle 28, he/she put again his/her hand inside the workspace. From that cycle we note that *Give* activations are increased until the task is completed (Cycle=31). Then, the operator delivers again the object to the robot (Cycle=38), the robot returns the object to the operator (Cycle=43) and the operator passes again the object to the robot (Cycle=49). Finally, the operator is no more interested in the object, so the *Place* behavior starts to increase its frequency of activation and the robot consequently releases the object on the table at Cycle 54.

7.5 Experimental Results or Evaluation Criteria

Evaluation Criteria To evaluate the performance of the attentional system and of the HRI system, we introduce some evaluation criteria considering safety, reliability, effectiveness, efficiency.

- *Safety* is measured in counting dangerous human-robot interaction events (i.e. a safe robot should avoid collisions between human and a moving robot and it should minimize interactions where the two are too close).

- *Reliability* is evaluated considering unrecoverable world/robot states encountered during the tests (the robot is stuck, the object falls down, the object is not reached or located by the robot).

- *Effectiveness* is assessed considering the time needed to achieve the task (the

system should minimize the time to achieve the task).

- *Efficiency* is associated with the number of behavior activations needed to achieve the task (for us, an attentional system is efficient, when it can distribute computational resources among different processes, focusing only on relevant activities).

Parameters Setting Given the attentional model introduced in the previous section, the overall attentional behavior is obtained once we tune the parameters associated with the behavior monitoring strategies.

To assess the system performance with respect to the previous set of criteria we introduced a suitable optimization function:

$$\begin{aligned} f = \quad & M1 \times N_{Safe} + M2 \times N_{Rel} + M3 \times T_{Effe} + \\ & M4 \times N_{Effi}. \end{aligned}$$

Here, $M_1 > \cdots > M_4$ specify the priorities in terms of weights; N_{Rel} represents the number of unrecovered situations with respect to the number of accomplished activities (pick, place, etc.); N_{safe} gives the HRI unsafe situations with respect to the executed activities; T_{Effe} is for the time spent to achieve the tasks with respect to the overall mission time; N_{Effi} is the number of behavior activations with respect to the maximal possible activations.

This function can be exploited, during the setting phase, to learn the system parameters and, during the testing phase, to validate the overall system behavior. Different learning algorithms can be deployed for parameter learning (e.g. genetic algorithms, particle swarm optimization, simulated annealing etc.). Currently, we are investigating the Differential Evolution (DE) algorithm [90], [91] which is particularly suitable for both unbounded and granular problems; indeed DE manages unrestricted and unbounded range of values.

Experimental Setup In order to evaluate the performance of the AIRM architecture, we compared it with a classical non-rhythmic architecture (P1Vmed) in which the behaviors perceptual schema are always active. P1Vmed is the baseline used to emphasize the advantage of our attentional mechanisms. The P1Vmed

regulation was deliberately simple: always active with a constant speed regulation which trades-off safety and efficiency. For the adaptive version (AIRM) we considered adaptive concurrent clocks with $p_{min} = 1$, $p_{max} = 10$ and $speed = \frac{max - speed \times p}{p_{max}}$ for all the behaviors. For the (P1Vmed), we assumed that the behavior perceptual schema are always active (i.e., p_{min} and p_{max} are both equal to 1) and the arm speed is set to a constant value ($speed = \frac{max - speed}{2}$). Moreover, in the case of the AIRM architecture, the updating policies of the behaviors are those specified in the previous section. The range of values for the speed is $[0, 0.3]$ m/s.

Empirical Results During the empirical evaluation, we tested each behavior 20 times with five different operators unaware of the robot behavior. Operators were required to simply observe the robot and move around in the case of Pick and Place behaviors, and interact, without any specific requirement, for the Give and Receive behaviors. In these final cases all the hand movements, made by the operators, were spontaneous. For each test we evaluated the parameters defined above: effectiveness, efficiency, reliability and safety.

	Reliability		Safety	
	AIRM	P1VMED	AIRM	P1VMED
Receive	1	1	1	1
Give	0.83	0.8	0.9	0.84
Pick	0.77	0.54	1	1
Place	1	1	1	1

Table 7.1: Evaluation of the Safety and Reliability criteria.

	Effectiveness		Efficiency	
	AIRM	P1Vmed	AIRM	P1Vmed
Receive	$7.66s \pm 0.54s$	$9.69s \pm 0.31s$	14.5 ± 1.57	41.7 ± 1.42
Give	$4.87s \pm 1.4s$	$7.27s \pm 2.9s$	6.05 ± 2.65	14.65 ± 5.59
Pick	$9.14s \pm 2.07s$	$10.48s \pm 0.67s$	16.2 ± 6.58	32.65 ± 5.99
Place	$6.03s \pm 1.05s$	$8.96s \pm 0.6s$	12.95 ± 5.03	58.65 ± 10.17

Table 7.2: Evaluation of the Effectiveness and Efficiency criteria.

More precisely, for the effectiveness we computed mean and standard deviation

of the time required to carry out the correspondent behavior; for the efficiency, for each behavior, we evaluated the mean and standard deviation of the number of behavior perceptual schema activations that are necessary to accomplish the task; for the reliability, we evaluated the percentage of trials in which the robot successfully completed the task towards the total number of tests carried out. Finally, for the safety evaluation we considered the number of times the robot collided with the operator and the number of times the robot did not stop its motion while the operator was within the robot proximity space with respect to the total number of tests in percentage.

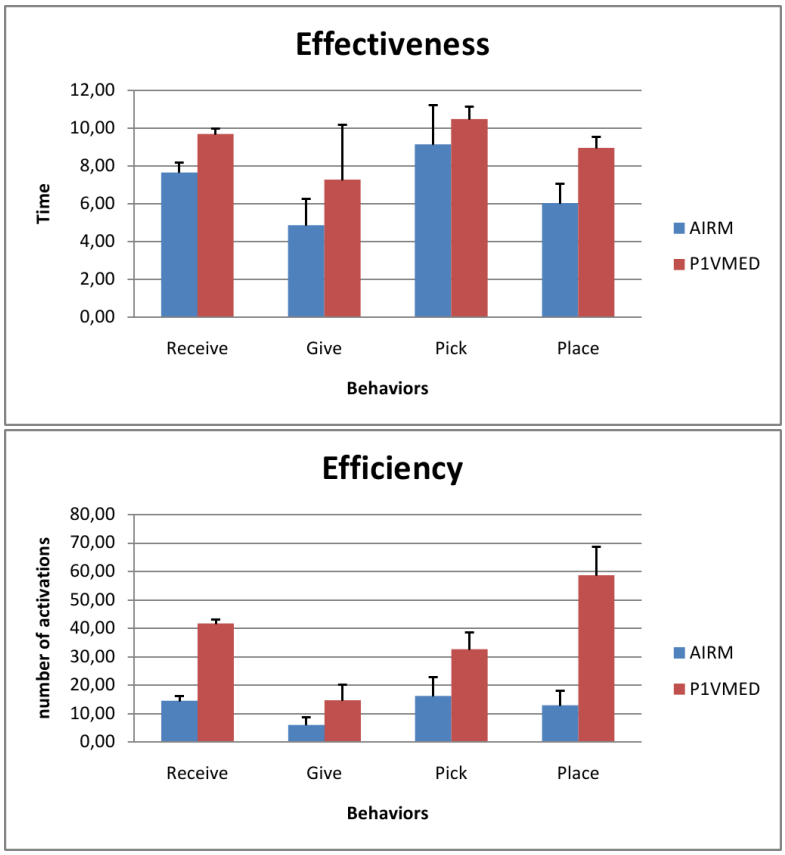


Figure 7.6: Effectiveness and Efficiency evaluation criteria.

Notice that, the attentional mechanisms are not only associated with better performance in terms of effectiveness and efficiency (Fig. 7.6 and Tab. 7.2), but we also observe better results regarding reliability and safety (Tab. 7.1), compared

with the non-adaptive architecture in which the behavior perceptual schemas are always active (P1Vmed).

In particular, notice that the adaptive modulation of the robotic arm speed allows us to accomplish the task faster than keeping the speed to a constant value, furthermore, the adaptive trajectory is safer and more comfortable from the operator point of view.

As we expected, a small number of activations has a big impact in the efficiency for the adaptive system.

Finally, the critical operations for the Safety and Reliability are the Give and Pick operations. As for safeness, the Give interaction is more critical (where the robot has to pass an object to the operator) than the Receive one (where the robot has to receive an object from the operator) causing more frequent collisions. The same happens for reliability, indeed, passing an object to a human is more difficult than receiving an object (note that our robotic arm has no force control on the end effector and its only relies on vision). Although in these cases the success rate is not equal to 100% (as in the cases of Receive and Place behaviors), the architecture endowed with AIRMs allows, due to its ability of adaptation, a number of successes larger than the P1Vmed standard architecture. For example, in the picking behavior the slower speed of the adaptive architectures permits a more accurate grip of the object.

7.6 Conclusions

The aim of this application was the specification of the attentional models employed in human-robot interaction. In particular we proposed a human-robot interactive system endowed with attention mechanisms used to coordinate simple manipulation tasks.

In the proposed attentional model, each behavior is equipped with an adaptive clock and an updating policy that changes the frequency of sensory readings (focusing the attention towards relevant movements of the operator the robotic arm interacts with) and modulates the emergent behavior in terms of variations of the robot arm speed.

We defined a simple control architecture for HRI considering pick-and-place

and give-and-receive attentional behaviors and to assess the system performance we introduced suitable evaluation criteria taking into account safety, reliability, efficiency, and effectiveness. Putting aside the efficiency parameter, that is a peculiar characteristic of an attentional executive systems, in our opinion the role of the attentional system is to trade off among safety, effectiveness and reliability in human-robot interaction and cooperation. A safe and reliable human-robot interaction means not only to stop the arm movements in dangerous situations, but also to modulate the arm speed during the interaction balancing faster movements (more productive and effective) in free space and slower ones during human-robot interaction. During the interaction, the robot has to balance when to follow the human hand to achieve collaboration (the desire to interact with people and manipulate objects, thus an attraction of moving forwards and close persons and objects) and when to stop, move away from the human or simply slow down its execution speed in order to not to hurt (the fear of hurting people, thus cautions, slow movements, intensive monitoring and a repulsion towards close persons). Such orchestration of attitudes emerges from the interaction of different behaviors (for example Give and Avoid) that works at different rates depending on the surrounding environment and the priorities of tasks.

Part VI

Epilogue

Chapter 8

Conclusions and Future Work

8.1 Conclusions

One of the main issues that is currently matter of research in the community studying cognitive and bio-inspired robotics is to make robots able to deal with highly dynamic environments in autonomous way. Namely, a robotic system should be able to continuously monitoring the surrounding environment, trying to achieve its goal and, in the meantime, it should be also able to cope with unexpected situations. In order to guarantee both these issues, the robotic control system needs to efficiently spending its limited sensorial and cognitive resources.

In this thesis we addressed the above problems by proposing an attentional monitoring system, capable to opportunely manage limited resources of a robotic system in monitoring unpredictable and dynamic environments. Attentional mechanisms applied to autonomous robotic systems have been proposed elsewhere, but mainly for vision-based robotics. In all these systems the mechanism of selective attention is used to support visual abilities to focusing attention only on salient stimuli in the external environment and discarding the information not relevant for robot current purposes. Conversely, we are also interested in artificial attentional mechanisms suitable for execution monitoring. Indeed, inspired by ethological and biological studies, attesting the role of attention in the control of action, we want to endow our robotic control system with an attentional mechanism capable not only to focus on salient stimuli, as it has been already proposed in other works, but

also on action execution control. Hence our aim is to implement both the selective and divided attention mechanisms to achieve the dual goal respectively to focus on relevant stimuli and to opportunely split shared resources among concurrent behaviors, producing the suitable actions.

Starting from a behavior-based executive system, we introduced simple attentional mechanisms by endowing each behavior with an adaptive internal clock that regulates the frequencies of sensor readings and action activations. We named this mechanism AIRM (Adaptive Innate Releasing Mechanism). Here, the process of changing the frequency of sensory readings is interpreted as an increase or decrease of attention towards particular behaviors and aspects of the external environment. Moreover, we introduced some mutual influence rules to ensure that the rhythm of competitive behaviors activations directly influences the rhythm of the associated behaviors. This influence can lead to synergistic or inhibitory activation mechanisms. In this setting, the overall attentional control is an emergent behavior obtained by the interaction of the monitoring strategies associated to each primitive behavior.

We investigated the feasibility of the use of adaptive internal clocks to implement these monitoring attentional mechanisms and, in order to validate our approach, we experimented the developed attentional control architecture in many different scenarios.

The results show that the realized attentive mechanisms are effective in adapting the frequency of behaviors activations according to the particular circumstances, incrementing or decreasing the attention towards salient aspects of the robot environment or the internal state and incrementing or decrementing the related behaviors activations. We showed that the so composed mechanism is able to filter the sensory information and split resources among different concurrent/-cooperative behaviors, adapting to the surrounding environment changes and to the internal needs of the robot.

The two main advantages introduced are that:

- being the activation mechanism periodic, it permits to reduce the number of behavior activations (as opposed to cases where the standard activations are performed every machine cycle), causing a relative decrease of computational load and thus improving the system performance;

- moreover, the adaptability allows the robot to move in safety, changing its reaction coherently to the specific environmental conditions. In particular it lets the robot able to read sensors more often if a dangerous situation occurs and less often in cases of a safe operational circumstance, showing in this way an *intelligent* behavior.

Basically, the system can modulate the activation frequencies on the basis of the available resources and external conditions. Indeed, by using the adaptive clocks, the number of behaviors activations substantially decreases compared to the case where the control system enables the robot behaviors at each machine cycle, and this results in a substantial gain in performances.

The results collected in all the test-beds also show that attentional mechanisms reduce the time to achieve the goal (effectiveness) and the activations of each behavior (efficiency), permitting a smooth and natural emergent behavior in all the considered scenarios, trading off between adaptivity and performances.

Furthermore, the experiments have also shown interesting results about the synchronization and the scheduling of the behaviors. In fact, since each behavior is endowed with its own clock, whose period can change over time basing on external and internal conditions, we have that, in some circumstances, a behavior is activated more frequently than other behaviors with the consequence that its influence on the emergent behavior is stronger than the others. This produces a kind of priority scheduling of the behaviors, that is completely decentralized and managed by each individual behavior, which adjusts and adapts its frequency, and consequently its priority value, increasing its weight in determining the output action.

We also observed that our first implementation of the AIRM architecture conveys the advantages of both the purely reactive and the deliberative architectures. Indeed it produces a quick reaction to the perceived stimuli as well as in the reactive architectures, but while in pure reactive system the stimulus response patterns are stereotypical, deterministic, and strictly dependent on the current state, in our frequency-based model this pattern is more complex: it is modulated by the behaviors' sampling and activation rates which are history-dependent.

We implemented the AIRM mechanisms by means of a particular neural network in order to face real time applications. We planned to use neural networks

also because they are ideally suited to the development of learning processes, since these represent a powerful data modeling tool, able to capture and represent complex input/output relationships. Indeed the motivation for the development of neural network technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain, where the knowledge is acquired through learning directly from the data being modeled.

Then in order to deal with more complex tasks we introduced a hybrid architecture, in which the attentional mechanisms are deployed in different layers of the structure. In this three-layer architecture the clock mechanism is indeed applied to the reactive layer, as well as in the preliminary implementation of the architecture, and to a planner with a variable planning horizon, in a way that the length of the planning horizon can be modified according to the particular environment conditions. This adaptability permits the control system to keep a coherent and finalized behavior while compensating failures and unexpected events.

In the design of all the different implementations of the attentive control systems we must, however, take into account both the external factors related to the environment in which the robots is situated and the internal composition of the system. Thus the choice of the maximum period of behavior activation, or the chosen monitoring strategy or the size of the planning horizon must therefore represent a fair compromise between the ability to reduce the number of activations, and thus reduce processing times unnecessary data, and the risk of not being able to quickly react to occurring environment changes. We demonstrated how evolutionary strategies can be used in a very fruitful way in order to learn the updating policies of the behaviors rhythms and make the system adaptable to different environments while improving its performance. We validate the efficiency of the system so learned proving its adaptability at different level of environmental complexity.

In the last part of this thesis we also show how our attentional architecture can be used in a wide range of applications, from the mobile robotics to human-robot interaction tasks, in order to ensure an effective improvement of the system performance and assure the execution of tasks in safety and reliability.

In this work, we have shown that it is possible to build control systems for

mobile robots performing better than the classic behavior-based architectures in terms of best use of resources, behaviors scheduling, and, in general, of effectiveness (reducing the time to achieve the goal) and efficiency (reducing the behaviors activations). In all the application contexts, in which these mechanism have been tested, we have shown that these attentional mechanisms can be used to ensure a real improvement in performance and in tasks execution in safety and reliability.

8.2 Further research topics

There are several development lines of the system realized. First of all, we aim to investigate the possibility to use both the evolutionary and online learning technics to improve the performance of the system. Indeed, in order to really make the robot able to adapt to very drastic environment changes, we think that the adoption of on-line learning could be very useful since it should make the robot able to detect modifications occurring during its own lifetime. In this regard, we are currently testing the use of an on-line learning algorithm, reinforcement-learning like, to compensate the evolutionary algorithm to the adaptation with respect to environment changes that are too fast for evolution to be tracked.

Another development line might also address the issue of porting of the attentional neural network on FPGAs (Field Programmable Gate Array) devices, which, given their advantages such as the reconfigurable logic, are currently the standard for the development of complex electronic boards. In this context, the NSL language (Neuro-Symbolic Language for Neuro-Symbolic Processors (NSP)) [93] through which describes the network and compilers that translate these statements in VHDL (VHSIC hardware description language - Very Hight Speed Integrated Circuits), makes the porting of the neuro-symbolic neural network on the FPGA devices almost immediately.

As a future work, we also plan to create an automatic and decentralized monitoring system for sensors networks management, in which the attentional monitoring system is directly applied to individual sensors or nodes. The purpose is to investigate whether even in this case, the rhythmic attentional systems can reduce the computational complexity, optimizing system resources and solving management problems of the computational load distribution on the sensors network,

without the aid of a centralized control, but relying only on the presence of *smart sensors* endowed with independent self-regulating mechanisms.

Bibliography

[1] D. Fox, W. Burgard, and S. Thrun, *2007 World Robot Market*, 2007. `http://www.ifrstat.org/downloads/2008_executive_summary.pdf`.

[2] M. Kuhnmunch, "Reality of robotics," *European Cleaning Journal*, 1997. `http://www.nada.kth.se/~hehu/robo/articles/future3/P_RealRobot.html`.

[3] iRobot website `http://www.irobot.com`.

[4] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[5] W. Zhao, R. Chellappa, P. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *Acm Computing Surveys (CSUR)*, vol. 35, no. 4, pp. 399–458, 2003.

[6] C. Galindo, J. Fernandez, and J. Gonzalez, "Hierarchical task planning through world abstraction," *IEEE Transactions on Robotics*, vol. 20, no. 4, pp. 667–690, 2004.

[7] R. A. Brooks, "Intelligence without reason," in *Proc. of the 12th IJCAI*, (Sidney, Australia), pp. 569–595, 1991.

[8] G. Miller, E. Galanter, and K. Pribram, *Plans and the Structure of Behavior*. New York: Holt, Rinehart, and Winston.

[9] R. Fikes and N. Nilsson, "Strips: A new approach to the application of theorem proving to problem solving," *Artificial Intelligence*, vol. 2, pp. 189–208, 1971.

[10] N. Chomsky, "Rules and representations," *Behavioral and Brain Sciences*, vol. 3, pp. 1–61, 1980.

[11] M. I. Posner and S. Petersen, "The attention system of the human brain," *Annual Review of Neuroscience*, vol. 13, pp. 25–42, 1990.

[12] M. Posner, "Orienting of attention," *Quarterly Journal of Experimental Psychology*, vol. 32, pp. 3–25, 1980.

[13] M. Posner and C. Snyder, "Attention and cognitive control," in *Information Processing and Cognition: The Loyola Symposium.*

[14] D. Norman and T. Shallice, "Attention in action: willed and automatic control of behaviour," *Consciousness and Self-regulation: Advances in Research and Theory*, vol. 4, pp. 1–18, 1986.

[15] D. Allport, "Visual attention," in *Foundations of cognitive science*, pp. 631–682, InM.I. Posner Ed. Cambridge, MA: MIT Press, 1989.

[16] E. Burattini, A. Finzi, S. Rossi, and M. Staffa, "Monitoring strategies for adaptive periodic control in behavior-based robotic systems," *Advanced Technologies for Enhanced Quality of Life*, pp. 130–135, 2009.

[17] H. Pashler, *The Psychology of Attention.* MIT Press, Cambridge, MA, 1998.

[18] J. Wolfe, *Visual Search.* H. Pashler, Hove, U.K.: Psychology Press, 1998.

[19] M. Chun and J. Wolfe, "Visual attention," vol. Ch 9, pp. 272–310, 2001.

[20] S. Frintropa, E. Rome, and H. I. Christensen, "Computational visual attention systems and their cognitive foundations: A survey," *ACM Trans. on Applied Perception*, vol. 7(1), 2010.

[21] N. D. B. Bruce and J. K. Tsotsos, "Saliency, attention, and visual search: An information theoretic approach," *Journal of Vision*, vol. 9, no. 3, pp. 1–24, 2009.

[22] C. Bundesen and T. Habekost, *Attention.* London: Sage Publications, 2005.

[23] R. Desimone and J. Duncan, "Neural mechanisms of selective visual attention," *Annual Reviews of Neuroscience*, vol. 18, pp. 193–222, 1995.

[24] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, 1998.

[25] J. Baccon, L. Hafemeister, and P. Gaussier, "A context and task dependent visual attention system to control a mobile robot," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 238–243, 2002.

[26] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," *Human Neurobiology*, vol. 4, no. 4, pp. 219–227, 1985.

[27] L. Itti and P. Baldi, "Bayesian surprise attracts human attention," *Vision Research*, vol. 49, no. 10, pp. 1295–1306, 2009.

[28] D. Gao, S. Han, and N. Vasconcelos, "Discriminant saliency, the detection of suspicious coindidences, and applications to visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 6, p. 17, 2009.

[29] A. Torralba, A. Oliva, M. Castelhano, and J. Henderson, "Contextual guidance of eye movements and attention in real-world scenes: The role of global features on object search," *Psychological Review*, vol. 113, no. 4, pp. 766–786, 2006.

[30] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97–136, 1980.

[31] J. Wolfe, "Guided search 2.0: A revised model of visual search," *Psychonomic Bulletin and Review*, vol. 1(2), pp. 202–238.

[32] T. Minato and M. Asada, "Image feature generation by visio-motor map learning towards selective attention," in *Proceedings of the IEEE/RSJ In-*

ternational Conference on Intelligent Robots and Systems, pp. 1422–1427, 2001.

[33] N. Mitsunaga and M. Asada, "Visual attention control for a legged mobile robot based on information criterion," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 244–249, 2002.

[34] A. Carbone, A. Finzi, A. Orlandini, and F. Pirri, "Model-based control architecture for attentive robots in rescue scenarios," *Autonomous Robots*, vol. 24, no. 1, pp. 87–120, 2008.

[35] J. Garforth, S. L. McHale, and A. Meehan, "Executive attention, task selection and attention-based learning in a neurally controlled simulated robot," *Neurocomputing*, vol. 69(16-18), pp. 1923–1945, 2006.

[36] S. Frintrop, P. Jensfelt, and H. Christensen, "Attentional landmark selection for visual slam," *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 2582–2587, 2006.

[37] C. Siagian and L. Itti, "Biologically-inspired robotics vision monte-carlo localization in the outdoor environment," *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 1723 – 1730, 2007.

[38] J.Garforth, S. McHale, and A. Meehan, "Executive attention, task selection and attention-based learning in a neurally controlled simulated robot," *Neurocomputing*, vol. 69, no. 16-18, pp. 1923–1945, 2006.

[39] G. Wasson, D. Kortenkamp, and E. Huber, "Integrating active perception with an autonomous robot architecture," *Robotics and Autonomous Systems*, vol. 26, pp. 325–331, 1999.

[40] D. Allport, "Selection for action: Some behavioral and neurophysiological considerations of attention and action," *Perspectives on Perception and Action*, pp. 395–419, 1987.

[41] S. Anderson, N. Yamagishi, and V. Karavia, "Attentional processes link perception and action," *Proceedings of the Royal Society of London Series B-Biological Sciences*, pp. 1225–1232, 2002.

[42] J. R. Simon, "Reactions toward the source of stimulation," *Journal of Experimental Psychology*, vol. 81, pp. 174–176, 1969.

[43] R. Nicoletti and C. Umilta, "Splitting visual space with attention," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 15, pp. 164–169, 1989.

[44] T. H. Stoffer, "Attentional focussing and spatial stimulusresponse compatibility," *Psychological Research*, vol. 53, no. 2, pp. 127–135, 1991.

[45] S. P. Tipper, C. Lortie, and G. C. Baylis, "Selective reaching: evidence for action-centered attention," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 18, no. 4, pp. 891–905, 1992.

[46] B. M. Sheliga, L. Craighero, L. Riggio, and G. Rizzolatti, "Effects of spatial attention on directional manual and ocular responses," *Experimental Brain Research*, vol. 114, no. 2, pp. 339–351, 1997.

[47] M. Tucker and R. Ellis, "On the relations between seen objects and components of potential actions," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 24, pp. 830–846, 1998.

[48] D. Kahneman, *Attention and Effort*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[49] T. Shallice, "Fractionation of the supervisory system," *Principles of the Frontal Lobe Function*, pp. 261–277, 2002.

[50] R. Cooper and T. Shallice, "Contention scheduling and the control of routine activities," *Cognitive Neuropsychology*, vol. 17, pp. 297–338, 2000.

[51] A. Stoytchev and R. C. Arkin, "Combining deliberation, reactivity, and motivation in the context of a behavior-based robot architecture," in *Proceed-*

ing of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, pp. 290–295, 2001.

[52] G. Pezzulo and G. Calvi, "A schema based model of the praying mantis," in *Proceedings of the 9th International Conference on Simulation of Adaptive Behavior*, vol. 4095 of *Lecture Notes in Computer Science*, pp. 211–223, Springer, 2006.

[53] G. Buttazzo, G. Lipari, M. Caccamo, and L. Abeni, "Elastic scheduling for flexible workload management," *IEEE Transactions on Computers*, vol. 51, no. 3, pp. 289–302, 2002.

[54] G. Beccari, S. Caselli, and F. Zanichelli, "A technique for adaptive scheduling of soft real-time tasks," *Real-Time Systems*, vol. 30, no. 3, pp. 187–215, 2005.

[55] L. Ward, "Attention," *Scholarpedia*, vol. 3, no. 10, p. 1538, 2008.

[56] D. Parisi, "Internal robotics," *Connection Science*, vol. 16, no. 4, pp. 325–338, 2004.

[57] E. Burattini and S. Rossi, "A robotic architecture with innate releasing mechanism," in *2nd International Symposium on Brain, Vision and Artificial Intelligence*, vol. 4729 of *Lecture Notes in Computer Science*, pp. 576–585, Springer, 2007.

[58] M. Staffa, S. Rossi, and E. Burattini, "Adaptive periodic control systems in robotics: A case study," *Verso la Robotica Intenzionale - The Tenth Meeting of the Italian Association of Artificial Intelligence AI*IA-2008*, 2008.

[59] C. Sherrington, *The Integrative Action of the Nervous System.* New York: Charles Scribners Sons, 1906.

[60] E. Adrian and F. Buytendijk, "Potential changes in the isolated brain stem of the goldfish," *The Journal of Physiolpgy*, vol. 71, pp. 121–135, 1931.

[61] P. Weiss, "Does sensory control play a constructive role in the development of motor coordination?," *Schweizerische Medizinische Wochenschrtft*, vol. 71, pp. 591–595, 1941.

[62] E. von Holst, "Die relative koordination als phnomen und als methode zentralnervser funktionsanalyse," *Ergebnisse der Physiologie*, vol. 42, pp. 228–306, 1939.

[63] E. Niebur, S. Hsiao, and K. Johnson, "Synchrony: A neuronal mechanism for attentional selection?," *Current Opinion in Neurobiology*, vol. 12, pp. 190–194, April 2002.

[64] T. Womelsdorf and P. Fries, "The role of neuronal synchronization in selective attention," *Current Opinion in Neurobiology*, vol. 17, pp. 154–160, 2007.

[65] A. Clark and R. Grush, "Towards a cognitive robotics," *Adaptaptive Behavavior*, vol. 7, no. 1, pp. 5–16, 1999.

[66] S. Treue, "Neural correlates of attention in primate visual cortex," *Trends in Neurosciences*, vol. 24, pp. 295–300, 2001.

[67] E. Burattini and S. Rossi, "Periodic activations of behaviours and emotional adaptation in behaviour-based robotics," *Connection Science*, vol. 22, no. 2, pp. 197–213, 2010.

[68] K. Lorenz, *King Solomon's Ring*. Penguin, 1991.

[69] N. Tinbergen, *The study of instinct*. 1951.

[70] A. Stoytchev and R. C. Arkin, "Incorporating motivation in a hybrid robot architecture," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 8, no. 3, pp. 269–274, 2004.

[71] W. L. Koukkari and R. B. Sothern, *Introducing Biological Rhythms*. Springer-Verlag, 2006.

[72] M. A. Arbib, "Schema theory," in *The handbook of brain theory and neural networks*, pp. 830–834, Cambridge, MA, USA: MIT Press, 1998.

[73] M. Atkin and P. Cohen, "Monitoring strategies for embedded agents: Experiments and analysis," *Adaptive Behavior*, vol. 4, no. 2, pp. 125–172, 1995.

[74] S. J. Ceci and U. Bronfenbrenner, "Don't forget to take the cupcakes out of the oven: Prospective memory, strategic time-monitoring, and context," *Child Development*, vol. 56, 1985.

[75] "Mobilerobots inc.," *http://www.mobilerobots.com/Mobile_Robots.aspx"*.

[76] B. Gerkey, R. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *proccedings of the International Conference on Advanced Robotics*, pp. 317–323, 2003.

[77] R.Wehner and S.Wehner, "Path integration in desert ants:approaching a long-standing puzzle in insect navigation," *Monitore Zool. Ital.*, vol. 20, pp. 309–331, 1986.

[78] R. Arkin, *Behavior-based Robotics*. Cambridge, MA: MIT Press, 1998.

[79] G. Rizzolatti, L. Riggio, and B. Sheliga, "Space and selective attention," *C. Umilta and M. Moscovitch (Eds.), Attention and Performance XV*, pp. 231–265, 1994.

[80] M. Matari'c, "Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior," in *Trends in Cognitive Science*, pp. 82–87, 1998.

[81] R. Arkin, K. Ali, A. Weitzenfeld, and F. Cervantes-Perez, "Behavioral models of the praying mantis as a basis for robotic behavior," *Robotics and Autonomous Systems*, vol. 32, no. 1, pp. 39–60, 2000.

[82] C. J. D. Patten, A. Kircher, J. Ostlund, and L. Nilsson, "Using mobile telephones: cognitive workload and attention resource allocation.," *Accid Anal Prev*, vol. 36, pp. 341–350, May 2004.

[83] J. L. Harbluk, Y. I. Noy, and M. Eizenmann, "Impact of cognitive distraction on driver visual behavior and vehicle control," tech. rep., 81st Annual Meeting of the Transportation Research Board, Washington, DC, January 2002.

[84] J. Coull and A. Nobre, "Where and when to pay attention: The neural systems for directing attention to spatial locations and to time intervals as revealed by both pet and fmri.," *Journal of Neuroscience*, vol. 18, no. 18, pp. 7426–7435, 1998.

[85] C. Miniussi, E. Wilding, J. Coull, and A. Nobre, "Orienting attention in time: Modulation of brain potentials," *Brain*, no. 122, pp. 1507–1518, 1999.

[86] E. Burattini and S. Rossi, "Periodic adaptive activation of behaviors in robotic system," *Internation Joutnal Pattern Recognition and Artificial Intelligence*, vol. 22, no. 5, pp. 987–999, 2008.

[87] M. Pattie, "A bottom-up mechanism for behavior selection in an artificial creature," in *Proceedings of the first International Conference on Simulation of Adaptive Behavior on From Animals to Animats*, (Cambridge, MA, USA), pp. 238–246, MIT Press, 1990.

[88] D. H. Ballard, "Animate vision," *Artificial Intelligence*, vol. 48, no. 1, pp. 57–86, 1991.

[89] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[90] E. Burattini, A. Finzi, S. Rossi, and M. Staffa, "Attentive monitoring strategies in a behavior-based robotic system: An evolutionary approach," in *Proceedings of the International Conference on Emerging Security Technologies*, pp. 153–158, 2010.

[91] M. Staffa, S. Rossi, M. De Gregorio, and E. Burattini, "Thresholds tuning of a neuro-symbolic net controlling a behavior-based robotic system," *Proceedings of the 19th European Symposium on Artificial Neural Networks, computational intelligence and machine learning*, vol. 73, pp. 159–164, 2011.

[92] E. Burattini, A. De Francesco, and M. De Gregorio, "NSL: A neuro-symbolic language for a neuro-symbolic processor (NSP)," *International Journal of Neural Systems*, vol. 13, pp. 93–101, 2003.

[93] E. Burattini, M. De Gregorio, and S. Rossi, "An adaptive oscillatory neural architecture for controlling behavior based robotic systems," *Neurocomputing*, vol. 73, pp. 2829–2836, 2010.

[94] S. Nolfi, "Learning and evolution in neural networks," 1990.

[95] R. Sutton and A. Barto, "Reinforcement learning: An introduction," *MIT Press.*, 1998.

[96] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[97] Y. Moron and G. Hayes, "Attention and social situatedness for skill acquisition," in *Proceedings of the first international workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotics Systems*, pp. 105–114, 2001.

[98] J. Garforth, S. McHale, and A. Meehan, "Problems of attentional behaviour in autonomous robotic systems," in *AAAI Technical Report SS-01-06*.

[99] R. Cooper and T. Shallice, "Contention scheduling and the control of routine activities," *Cognitive Neuropsychology*, vol. 17, pp. 297–338, 2000.

[100] E. Burattini, S. Rossi, A. Finzi, and M. Staffa, "Attentional modulation of mutually dependent behaviors," in *Proc. of SAB-2010*, pp. 283–292, LNAI 6226, 2010.

[101] D. Wilkins, "Practical planning: Extending the classical AI planning paradigm," *Morgan Kaufmann Series in Representation and Reasoning*, 1998.

[102] M. Wooldridge, "Intelligent agents," *G. Weiss (edt.) Multiagent Systems*, pp. 22–77, 1999.

[103] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. London: Springer, 2008.

[104] T. Estlin, G. Rabideau, D. Mutz, and S. Chien, "Using continuous planning techniques to coordinate multiple rovers," *Workshop on Scheduling and Planning Meet Real-time Monitoring in a Dynamic and Uncertain World*, 1999.

[105] R. Bonasso, R. Firby, E. Gat, D. Kortenkamp, D. Miller, and M. Slack, "Experiences with an architecture for intelligent, reactive agents," *Proceedings of the International Joint Conferences on Artificial Intelligence*, 1995.

[106] P. Nayak and B. Williams, "A reactive planner for a model-based executive," *Porceedingns of the International Joint Conference On Artificial Intelligence*, 1997.

[107] E. Sisbot, A. Clodic, R. Alami, and M. Ransan, "Supervision and motion planning for a mobile manipulator interacting with humans," in *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, pp. 327–334, 2008.

[108] E. Sisbot, L. Marin-Urias, X. Broquere, D. Sidobre, and R. Alami, "Synthesizing robot motions adapted to human presence - a planning and control framework for safe and socially acceptable robot motions," *International Journal of Social Robotics*, vol. 2, no. 3, pp. 329–343, 2010.

[109] A. Edsingera and C. C. Kemp, "Human-robot interaction for cooperative manipulation: Handing objects to one another," in *IEEE Intarnational Symposium on Robot and Human Interactive Communication*, pp. 1167–1172, 2007.

[110] R. Alami, A. Albu-Schaeffer, A. Bicchi, R. Bischoff, R. Chatila, A. Luca, A. De Santis, G. Giralt, J. Guiochet, G. Hirzinger, F. Ingrand, V. Lippiello, R. Mattone, D. Powell, S. Sen, B. Siciliano, G. Tonietti, and L. Villani, "Safe and dependable physical human-robot interaction in anthropic domains: State of the art and challenges," in *Proceedings IROS Workshop on pHRI - Physical Human-Robot Interaction in Anthropic Domains*, 2006.

[111] C. Breazeal, *Designing Sociable Robots*. Cambridge, MA, USA: MIT Press, 2002.

[112] Y. Nagai, K. Hosoda, A. Morita, and M. Asada, "A constructive model for the development of joint attention," *Connection Science*, vol. 15, no. 4, pp. 211–229, 2003.

[113] L. Wang and C. Chen, "A combined optimization method for solving the inverse kinematics problem of mechanical manipulators," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 489–499, 1991.

[114] P. Viola and M. Jones, "Robust real-time face detection," *International Journal on Computer Vision*, vol. 57, pp. 137–154, 2004.