Proceedings of International Workshop on Data Mining and Artificial Intelligence (DMAI' 08)
24 December, 2008, Khulna, Bangladesh

Behavior-based Robotics And The Reactive Paradigm

A Survey

L. De Silva[1] and H. Ekanayake[2]
University of Colombo School of Computing
35, Reid Avenue, Colombo 07
Sri Lanka
lalindra84@yahoo.com[1], hbe@ucsc.cmb.ac.lk[2]

Abstract

Behavior-based robots have come under the spotlight in recent years by making their presence known in diverse fields of human interest. Applications in behavior-based robotics have continued to grow in areas such as demining, search and rescue, office automation, health care, etc and continue to replace human beings in risky and menial tasks. With this inspiration, this paper investigates a variety of aspects in behavior-based robotics and the reactive paradigm in the context of their origins, concepts, applications and current research and is intended to provide a comprehensive overview about this area in robotics. The paper will review several central issues including a brief history of robotics, transition of robotic systems from hierarchical paradigm to reactive paradigm, biological/ethological inspiration for behavior-based robots, fundamentals of the reactive paradigm, architectures for controlling robotic behavior and the hybrid deliberative/reactive paradigm.

1. Introduction

The human desire to mechanically model living creatures dates back to pre-Aristotle era (before 350BC), as far as the times where a Greek mathematician, Archytas, is thought to have built a mechanical bird, the first mechanical artifact of the sort [1]. History reveals a large number of examples where humans have attempted to build animal-like artifacts and instances like Leonardo's robot (a knight-like mechanical artifact designed by Leonardo Da Vinci around 1495AD) are mere examples of a larger lot [1]. Thousands of years of such experiments have led us to the present era in which we deal with cognitively advanced robots.

The field of behavior inspired robotics began shaping in the 1950s with the inspiring work done by Grey Walter with his invention of the electronic tortoise, the first robot to display reactive behavior [7]. Even without any knowledge of its environment or any representation of its world state, the electronic tortoise was able to produce reactive behavior by responding to various forms (the intensity) of light in different ways. Such inventions gave way to a more structured approach to design and implement robots, namely, the hierarchical paradigm. Section 2 covers in detail the characteristics of the hierarchical paradigm and why, after few decades of prevalence, it gave way to the reactive paradigm.

The extent of inspiration from ethological studies in shaping the reactive paradigm has been immense. Thus, section 3 delves into identifying the similarities between animal behavior and robots that are built on reactive behavior. Section 3 also considers how animal behavior relates to perception and action, how multiple behaviors are controlled and coordinated in animals and how schema theory can be adopted to represent these behaviors. Section 4 catalogs the reactive paradigm in detail and discusses how it emerged from ethological studies and became the leading underlying architecture for robotic systems for many years ahead. Several other related topics including how this paradigm deviated from the hierarchical paradigm, its prominent methods for controlling behaviors, the mechanisms to integrate competing behaviors and the shortcomings of this paradigm will be discussed in detail.

Section 5 describes the hybrid deliberative/reactive paradigm. This paradigm addressed the shortcomings of the purely reactive systems by incorporating the planning element into the methodology and has been identified currently as the most favorable architecture to design and implement the robotic systems. A large number of control architectures that are built upon the hybrid paradigm have appeared overtime and section 5 will take into account some of these architectures in order to exemplify the use of the hybrid paradigm.

2. Hierarchy Matters

2.1. The Unseparable Trilogy

In designing the architectures for robotic control, the earliest robotic systems used a high level structured approach known as the hierarchical paradigm. In systems that followed this paradigm, the robots followed a strict sequence of distinct states known as the SENSE, PLAN and ACT cycle [19]. In the SENSE stage, the robot would acquire information about its environment using its sensors and use this information to create a representative data structure, known as the world model. According to Murphy (2000), the typical world model would comprise of priori information of the environment, sensing information from the robot's sensors and any other cognitive knowledge that would assist the robot in carrying out the specified task. The PLAN stage receives this world model and devises an action plan for achieving the task. In cases when the final goal comprised of complex situations and world states, the planner would break the goal into sub goals and account for each in turn to achieve the final goal. The ACT stage represents the execution of actuator commands that are generated according to the directives received from the planning stage. When the robot's actuators have carried out the task specified, the cycle begins again and continues until the goal is reached.

Among the representative architectures and methodologies that were built on the hierarchical paradigm, the strips method was a forerunner and was used in Shakey, a robot by the Stanford Research Institute [5]. Strips method measured the difference between the goal state and the current state and chose the best action that would minimize the difference. Constructing the world model was imperative and the action to be chosen at a certain point was selected from a descriptive table called the difference table in which differing actions corresponding to varying differences were listed beforehand. During the planning stage of each cycle a difference-evaluator would measure the difference between the current state and the goal state, enabling the planner to select the corresponding commands from the difference table and pass them on to the actuators.

2.2. Two Challenging Problems

Among the biggest issues that made the hierarchical paradigm less inspiring with time were the close world assumption and the frame problem [19], [9]. The close world assumption assumes that the robot has obtained all the information it needs about the environment within the world model. Such an assumption presents a significant problem when the environment changes with time since the planner cannot keep track of all the changes of the environment in a continuous manner.

Hierarchical paradigm also suffered what is known as the frame problem [6]. Frame problem is the inability to represent all the world information that was needed by the robot in a computationally viable method. The required intricacy and the detail of the information of the robot's environment were considerable even for the robot that was meant to achieve the simplest task. Consequently, addressing uncertainty in the event of a bigger problem was too tedious and was not worth the effort.

2.3. Relaxing The Hierarchies

The nested hierarchical controller (NHC) proposed by Meystel (1986) suggested a method to resolve the aforementioned issues by enhancing the planner by decomposing it into three distinct components, namely, the mission planner, navigator and the pilot [18]. NHC's most obvious deviation from a pure hierarchical architecture was that the sensors would continuously update the world model while the actuators were carrying out the commands. This gave a more realistic response in terms of the receptiveness to the changes in the environment.

A similar architecture was proposed by Albus through his real-time control system and was an architecture that was heavily influenced by the NHC architecture [8]. Shanahan presented a logic-based high level robotic control approach which also used a hierarchical approach to control the robot's behavior [20].

Even with these proposed strategies to enhance the hierarchical paradigm, the fact that this paradigm fails to address uncertainty has made people look for different paradigms for robotic control. Thus, with the advances in the field of artificial intelligence, cognitive psychology and ethology, people turned to studying animal behavior and understanding how behavior coordination is handled biologically.

3. Animals To The Rescue

In the early days of robotics, the notion of using animal behavior as models in developing robotic systems was treated with much detest due to several reasons. The strongest argument against using animal behaviors was the inadequate knowledge of the intricate biological intelligence that was available at the time. Secondly, the available technology at the time was hardly sufficient to model even the simplest animal behaviors that had been identified.

However, the advances in cognitive psychology and ethology combined with the growing limitations faced in the hierarchical paradigm gave the roboticists an incentive to return to these biological studies and to search for new paradigms for robotic control. Still, the findings in these areas were not comprehensive enough to produce a model for robotic behavior that would directly map to a biological behavior. As a result, roboticists tried to come around this issue by going in the reverse direction, i.e. to build models for robotic behavior and to find confirmation of those behavioral models in biological and ethological findings. Furthermore, animals interact with a much more relaxed, open environment and thus the study of their behaviors provided a foundation to tackle the close world assumption which proved to be the biggest obstacle in the hierarchical paradigm [19].

3.1. The Basic Building Block

The foundation of behavior-based architecture for robotics lies in the definition of a behavior. In its simplest form, a behavior is a mapping of sensory inputs to a pattern of actions that in turn carries out a specific task [19]. Animal behaviors have been studied in three broad categories, namely, reflexive behaviors, reactive behaviors and conscious behaviors [19]. Reflexive behaviors are direct stimulus-response mappings in which the response to a particular sensory input is directly wired with a response and which is carried out without any cognitive involvement of the subject. An example for this type of behavior can be the reaction of a person's knee when it is tapped. The second category concerns with the behaviors that are learned over time and are ultimately embodied in the system such as learning to ride a bicycle or learning to walk in early childhoood. Every other behavior that requires conscious thought throughout the time of the action is categorized as conscious behavior.

In behavior-based robotics, reflexive behavior prevails over any other behavior type. Therefore, this paper will confine only to the reactive systems and will describe in detail how reactive behavior is introduced in robotics. However, there is a slight difference in the connotations between robotic literature and ethology. The word 'reactive' actually means reflexive behaviors in robotics while in other disciplines the word 'reactive' refers to the learned behaviors.

3.2. Behavior Diversity

In studying the reflexive behavior of animals, ethologists have been able to identify three forms of reflexive behaviors [19]. These forms can be explained as follows.

Reflexes: Where the magnitude of response is directly proportional to the intensity of the stimulus and lasts as long as the stimulus lasts. This form of behavior is visible in domestic millipedes and certain snails where they would contract their muscles in response to a disturbance [9].
Taxes: Where the response is an orientation towards or away from the stimulus. The best examples for this kind of behavior is the Chemotaxis [2] behavior of trail following ants and the behavior of sea turtles who would head directly to the sea once they are born. It has also been discovered that certain animals react to the magnetic field of the earth and display behavior accordingly [4].
Fixed Action Patterns: Where the response lasts for a longer time than the stimulus and where the behavior can be influenced from multiple stimuli rather than from a simple stimulus [3]. Examples are drawn for this sort of behavior in animals fleeing away from predators, the singing of crickets and locust fly patterns [9].

It should not be misunderstood from the above description that a pair of stimulus and response is all that needs in reflexive behavior. A mechanism to release the behavior in the presence of a stimulus should also exist in order to execute the behavior.

Innate Releasing Mechanism (IRM) was a phrase coined by Lorentz and Tinbergen (1973) to denote the mechanism by which a specific behavior is initiated [9]. The underlying explanation is that a behavior will not be initiated even in the presence of sensory inputs unless some mechanism was present to initiate that behavior. One of the simplest examples from the animal world is that most predators would not hunt prey unless they are hungry, even if the prey wanders close by.

A releaser, which can be presumed to be a boolean value, has to be set in order to initiate the behavior. In most occasions, compound releasers, probably both internal and external, have to be present in order to trigger a behavior. This essentially means that IRMs act as control switches for releasing behaviors and provide a low level coordination mechanism for triggering and inhibiting basic behaviors.

3.3. Controlling Behaviors

Another advantage of studying animal behavior is the ability to understand how the concurrent, competing behaviors are handled biologically. Ethologists have identified several biological mechanisms that animals portray when they are faced with competing behaviors. These mechanisms that make choices between competing behaviors can assist roboticists in modeling robots with concurrent behaviors and several of such control mechanisms are listed below.

Equilibrium: Occurs when a dominant behavior cannot be identified and no behavior takes precedence over the other behaviors. Murphy (2000) illustrates the example of a squirrel on a park bench that sights a bread crumb near a person sitting on the edge of the same bench [19]. Most times it can be seen that the squirrel seems to be lost in a dilemma whether to go for the bread crumb (execute the feed behavior) or to flee from the situation (execute the flee behavior).

Dominance: Occurs when one behavior clearly dominates the others. The obvious example is that animals cannot eat and sleep at the same time.

Cancellation: Occurs when the competing behaviors cancel each other out so that a completely new behavior is displayed. Murphy (2000) draws an example from male stickleback fish. Sticklebacks get caught up in between defending their nests and attacking the enemies when their territories are threatened and end up building a new nest altogether in a different place [19].

3.4. A Novel Representation

Perception is a crucial factor in animal behavior as it plays a vital role in coordination and selection of different behaviors. Arkin (1998) identifies that perception can be two-fold at the highest level: Proprioception, which refers to perceptions arising from stimuli that are internal to the animal's biological system (For e.g. hunger) and exteroception that relates to perceptions arising from external stimuli such as the cue of a predator nearby [9].

An interesting theory presented by Gibson (1979) that has assisted the way roboticists think in terms of behaviors is the theory of affordances [15]. Affordances differ from stimuli in that affordances take into account the environment in which the stimuli was produced. A better explanation is that an affordance is a potentiality for inherent action [19], [9]. This analysis of perception and affordances leads to the intriguing issue of representing this information in robots. One successful way around this problem was presented by Arbib (1992) through his famous schema theory. Schema theory facilitates the representation of perceptual and motor-action information in a similar way to that of object oriented concepts. A schema is an entity comprising of perceptual data and computational activity that result in motor actions. Accordingly, a behavior is defined as comprising of perceptual schemas and motor schemas, one from each at a bare minimum [21]. Schema theory also backs the concept that complex behaviors can be comprised of much simpler behaviors, thus facilitating the behavior subdivision in robotic systems.

3.5. From Animals To Robots

Several concrete implementations stemming from these studies have been surfacing over the years and it appears necessary that a few of those examples be drawn out here. The chemotaxis display of ants has been modeled using robots that can sense a chemical trail left by other robots and these experiments have been a success in the study of multi-robot environments and how behavior coordination is handled in such environments [9]. The locomotion patterns of the insects such as cockroaches have provided the substance to another research area and this area has concentrated on aspects such as path following in robots and other navigational behaviors. The artificial insect project gives examples for such experiments in great detail [13].

4. Planning: To Do Or Not To Do

The highlight of the reactive paradigm was its basic building block - a behavior [9]. As mentioned earlier a behavior is a direct mapping of sensory inputs to the corresponding motor actions. The biggest difference the reactive paradigm had introduced from the earlier hierarchical paradigm is the removal of the planning stage from the Sense-Plan-Act triad. This was due to the popular belief among the roboticists that maintaining world state, history information and planning of actions, no matter how efficiently designed, only adds to the deterioration of the performance of the system. Thus, the reactive paradigm relied on behaviors and the fact that emergent behavior of the system can be produced by different mechanisms that are available to coordinate these behaviors. The concept of a behavior facilitated solid architectures that allowed extension of older behaviors, inhibition of behaviors, parallel execution of behaviors and many other coordination mechanisms that produced the emergent behavior of the complete system [19].

Two other important considerations in behavior-based robotics are the local sensing techniques and the ecological niche. In the case of sensing, reactive paradigm advocates the use of a specific sensor input for each behavior. This effectively means that each behavior takes sensory inputs from a sensor that is local to it and passes this information to the perceptual schema. However, latter systems were built with architectures that can share sensory information where several behaviors would use sensory information from a single sensor.

The other important consideration in reactive robotics is the concept of an ecological niche. An ecological niche, from a purely biological point of view, means the environment in which the animal lives in and relative to which its behavior can be defined. In terms of robotics, this means that the roboticists should design the robot targeting a specific environment since the robot's interaction in a different environment would be meaningless and would not result in the expected outcomes.

4.1. Status Representation

In developing reactive architectures, several representational mechanisms have been introduced [19], [9]. Some of these representation mechanisms can be listed as follows.

Stimulus-Response diagrams: Presents a very primitive way to represent the collection of behaviors that are coordinated. This mechanism simply portrays each behavior as a mapping of sensory stimuli and the action response.
Functional Notation: Presents the behaviors in a procedural manner that is similar to a pseudocode of a general program. Murphy (2000) explains a similar method named scripts that functions as a general script of a play [19].
Finite State Acceptors: Presents a general approach that is used in many other areas in computer science that is used to express the situational information of a system. A finite state automata is defined as a tuple $M = \{Q, \Sigma, \delta, q0, F\}$ with the symbols representing the set of states, the input alphabet, the transition function, the start state and the set of acceptable final states respectively. This definition is best used in its graphical form and presents a clear method of understanding the interaction of different behaviors.

4.2. Building Upon The Reactive Paradigm

Behaviors by themselves are of little importance to a reactive system unless they are coordinated to produce the emergent behavior of the whole system. Several architectures have emerged over the years and have presented varying approaches to coordinate behaviors in robotic systems. This paper focuses on the two most tried architectures of all, namely, subsumption and potential-fields methodology.

In 1986, Brooks presented an inspiring paper titled "Intelligence Without Representation", in which he questioned the viability of representing the world state and planning the sequence of actions in robotic systems [11]. He vehemently questioned the practicability of using specific data structures and attempting to model the robot's environment. Instead, he stressed that 'the world is its own representation', i.e. that any other representation of the world that was to be used in the robotic systems would be futile since no model can perfectly account for the uncertainties of the real world.

Consequently, Brooks (1987) presented an architecture for controlling and coordinating the robotic behaviors, for which he gave the name, subsumption. Subsumption builds on layers of competencies (or simply, a collection of behaviors that achieved a specific task) that are vertically arranged in the order of their sophistication. Thus, the basic survival behaviors such as collision avoidance are aligned in the lowest layer while more cognitive behaviors such as path mapping are aligned in the higher levels [19]. The higher layers have the ability to inhibit or suppress the behaviors of the lower layers, hence the word, subsumption.

The layers of competences are supposedly executed independently with each behavior having its own sensorial system and each behavior being unaware of what's happening in the other layers. Each layer executes a dedicated behavior and this avoids the need for them to know the complete scenario they are trying to solve.

However, this separation of layers calls for a mechanism to control and coordinate the behaviors when several of them are trying to execute simultaneously. Subsumption resolves this issue by always giving the higher layer the ability to take control and override the behaviors of the lower layers. Allowing such superimposing behavior requires that the higher level behaviors embody the competencies of the lower layers.

Subsumption does not come without its shortcomings. First, the inclusion of lower-layer-competencies in higher levels means a waste of resources. Second, there often arises the requirement to pass information between layers and pure subsumption architecture acts in contrast to this by precluding the layers from passing information between themselves [24]. Finally, the subsumption architecture fails to take into account any advantage a planning module can introduce to the system.

Several solutions to address these rigidities of the subsumption architecture were proposed by many and they introduced several alterations to the original architecture. One such proposition (Flanagan et al, 1995) brought in three variations to the original architecture [24]. This method substituted the competences of the subsumption architecture by individual processes with each process carrying out a defined task. The methodology also added a global data structure where each process can post and read interesting data to and from this data structure. Finally, they included a priority scheme that allowed the behaviors to compete for control and made this priority scheme time varying so that no single behavior can hold up the robots control for a long time.

Yongjie, Qidan and Chengtao (2006) presented another modification to the subsumption architecture by assuming a hybrid control approach that was based on subsumption [12] (hybrid control architectures will be discussed in detail in section 5). In addition to the use of processes as in the previous approach, this methodology also introduced a behavior-managing layer. The task of the behavior-managing layer was to adjust the behavior parameters (for e.g. priorities, behavior arithmetic etc) according to a number of factors that effected the robotic system at a certain point in time. Finally, they added a scheduler to the system to select which behavior process will take control of the system and provide the necessary actuator command.

No matter how hard it is tried to reduce the rigidity of the purely reactive subsumption architecture, there always is a tradeoff in performance when planning and its constituent data structures are introduced to the system. However, this tradeoff will be justified in complex systems where purely reactive systems are insufficient to provide the complete behavior.

Another approach taken by the proponents of the reactive paradigm is a methodology known as the Potential Fields Methodology (PFM). PFMs build on the groundwork that each behavior is represented as a vector and thus the methodology is inherently regarded to be confined to the navigational robots. These behaviors, represented as vectors, are combined in vector summation to produce the emergent behavior.

For presentational purposes, these behaviors are assumed to exert on the robot in the form of force fields by the objects in the environment and are illustrated as magnetic or gravitational fields. For e.g. an obstacle in the environment may be illustrated as emitting a repulsive field from its vicinity as much as a bulb is radiating light whereas a goal in the environment can be illustrated by an attractive field on the surrounding area. In representing these force fields, the general representation of the vectors, where the magnitude of the vector is given by the length of the arrow and the orientation given by the angle, can be used (For e.g. in the case of a navigational robot, the length of the vector represents the velocity and the angle represents the direction of the robot). The robot is assumed to be a particle entered into the force field area and the behavior of the robot is the path the robot takes as a result of the multiple potential fields [19]. Magnitude profiles are used to control the strength of the potential field to avoid the unpleasant outcomes caused by having a constant strength throughout the field (For e.g. to avoid over shooting a goal, the field will be calculated such that the strength of the field is a reducing function of the distance between the robot and the goal. As the robot moves closer to the goal, the magnitude of the force reduces so as to allow collision-free arrival at the goal).

One common misconception is that the PFM requires the calculation of the entire field in one occurrence. However, this is not the case. The robot only needs to calculate the force fields that exert on it in each Sense-Act cycle [19]. Thus the computational load that would appear to exist in designing the complete force field at once is greatly reduced.

The greatest drawback of the PFM is the possibility of local minima existing in the force field environment. A local minimum occurs where the strength and orientation of the vectors cancel each other out so that the robot caught up in such a situation will be stuck there forever. Masoud (1996) proposed a way around this issue by introducing Harmonic Potential Fields [17] while Iianmiao and Bo (1992) introduced time-varying potential fields [9]. Arkin (1987) suggested an approach where random noise can be injected to the fields to shake off the robot out of a local minimum trap [9].

Koren and Borenstein (1991) draw a detailed mathematical analysis of several other limitations and drawbacks of the PFM and proposed an altogether different methodology for obstacle avoidance in navigational robots in real time environments [10].

Reactive paradigm emerged as the ultimate solution for the rigidities put forth by the hierarchical paradigm. Though this newly found paradigm proved more than satisfactory in robots executing simple tasks, the obvious need for planning in more complex tasks emerged with time. Hence, roboticists looked into new ways of combining the planning process with the reactive behavior of the robots and thus a new breed of architectures under the name Hybrid Deliberative/Reactive paradigm was born.

5. One Step Ahead: The Hybrid Deliberative/Reactive Paradigm

5.1. The Need For Planning

The reactive paradigm started showing its limitations when moving into domains with complex tasks that required cognitive capabilities rather than just reactive behavior. These requirements were presented in the form of efficient path planning, map making and performance evaluation of the robot itself, etc, and the reactive paradigm proved incapable of satisfying these requirements.

The proponents of the hybrid architecture advocate the use of the advantageous aspects of each of the previous two architectures and combining them to produce the desired architecture for the complex scenarios. Effectively, this meant that the planning aspect of the hierarchical paradigm was to be integrated with the rapid execution capabilities of the reactive paradigm. The technological foundations for such an architecture was emerging as well and these technological aspects facilitated such integration in the two paradigms (For e.g. the emergence of multithreaded architecture allowed the planning process to be executed simultaneously with the reactive behaviors but as a separate module from that of the reactive modules).

5.2. Integrating Plans

The hybrid paradigm altered the Sense-Plan-Act trilogy by allowing the Plan stage to be executed separately from the Sense-Act sequence. Arkin (1998) mentions three broad methods for integrating planning with reactive behavior [9]. Hierarchical integration makes the planning layer and the reactive layer to be aligned vertically in which the planning layer provides the information on which the reactive layer acts. Second method involves planning layer working prior to or concurrently with the reactive behaviors, updating the robots behavioral parameters or changing the world state.

Finally, coupled planning-reacting method makes the planning and reactive activities to occur concurrently, allowing each one to interact with the other in real-time.

Another interesting deviation of hybrid architectures from the reactive paradigm is that the sensors that were unique to each behavior in the reactive paradigm are now shared with the planning modules. Planning needs to create world models and to represent the state of the environment and thus the use of sensorial information is imperative.

5.3. The Hybrid Features

Though the architectures that fall under the hybrid paradigm vary considerably in their designs, several common features can be identified in most of them [19], [9].

Sequencer: The task of the sequencer is to generate the set of behaviors that should be executed and their sequence in order to carry out a specific task. A dependency network or a finite state machine is generally used in coordinating these behaviors.
Resource Manager: Introduction of this module was primarily due to the fact that resources such as sensors are now been shared between the planning and reactive modules. Resource manager handles the allocation of sensors and other resources between different modules and selects the best resource that should prove practical for each module.
Cartographer: The cartographer module's main application is to create, store and maintain maps of the robotic environment by interacting with the external world. This module facilitates the knowledge representing scheme of the robot by maintaining and updating the data structures pertaining to the world model.
Mission Planner: This module's task is two fold. Initially, it presents an interface to the outer world and secondly it transforms the external commands into robotic terms, subdivides the tasks and forwards the subtasks into actuator modules to carryout.
Performance Monitor: Robot's performance in executing a given task should be evaluated in order to verify if the robot is carrying out its tasks as it should be. The performance monitor measures the robots performance in comparison to previously set targets by the mission planner and prompts any unfavorable activity.

5.4. Hybrid Architectural Styles

Murphy (2000) draws out three architectural styles that are visible in the hybrid paradigm.

Managerial Style architectures pose the general managerial style that is evident in an organizational hierarchy by subdividing the tasks in each module and allowing each task to be carried out by a subordinate module. Secondly, State Hierarchies style uses the robot's state to determine the appropriate action. The reactive behaviors are thought to maintain no state while the deliberative modules keep track of the world model and the past actions of the robot. Finally, Module-Oriented styles follow architectures that closely resemble the ones in hierarchical paradigm in which most modules have access to the world models individually.

5.5. Representative Architectures

In discussing the varied architectures that were based on the hybrid paradigm, Autonomous Robot Architecture (AuRA) proposed by Arkin (1998) draws special interest. In this architecture the previously mentioned five models of a hybrid paradigm were clearly visible and the architecture had clear division between the deliberative and the reactive layers.

Stenzel (2000) presents a utility-based approach to handle the problem of combining commands of different behavior mechanisms [23]. Additionally, his approach uses a priority management system to asses the importance of goals and this system uses subsumption to inhibit the low priority goals giving way to the high priority ones. Egerstedt and others (1999) used hybrid automata to depict each behavior of the robot and allowed the behaviors to fuse rather than to allow one behavior to overpower the other [16]. Li and Feng (1994) proposed a fuzzy logic based approach to coordinate the behaviors of navigational robots. In this approach, they used fuzzy reasoning to weigh the reactive behaviors rather than to inhibit the lower level behaviors as in the subsumption process [14]. A similar behavior-based neuro-fuzzy controller was introduced by Rusu and others (2003) where the coordination system was organized in a top-to-bottom hierarchy and in which command fusion using neuro-fuzzy subsystems were used to handle multiple low level behaviors [25].

A comprehensive study of behavior coordination mechanisms in behavior-based robotics has been presented by Pirjanian (1999) and this paper covers in detail all the alternative methods in addition to the primary methods discussed above [22].

6. Conclusion

The reactive paradigm answered many problems that were persistent in the early robots due to the rigidity that was present in the hierarchical paradigm. Robots that were based on reactive behavior proved much more efficient and showed rapid execution in performing its specified tasks.

However, the inability to cater into much more complex task domains has made the pure reactive behavior-based robots to decline in popularity with the emergence of the hybrid paradigm. The obvious need for a planning module to provide the necessary information required to handle the complex behaviors and complex environments has made the latter paradigm much popular towards the recent years.

Even the inclusion of the planning element does not make the robots as intelligent as we would expect them to be. It may be true that a robot is able to find the most efficient path to a goal, to measure its performance, to produce efficient representations of the environment, etc but in order to reap the real robotic performance, a learning process has to be integrated to the system. The field of machine learning has been producing a lot of results that cater to this need but the extent to which the robotic world has made use of these mechanisms is still questionable.

Research areas such as cognitive modeling has been devising models to produce intelligent behavior in machines at a conceptual level but the implementation of such models in robots still seems to be at a very rudimentary level. However, it can be expected that with the advancements of such research which try to model human intelligence in machines could take the behavior-based robotics to the next level, a level in which the robots can cope up with the uncertainties of its environment at a highly acceptable level.

Therefore, the future that we are looking at in the context of behavior-based robotics, to a large extent, would focus on hybrid architectures with robotic learning playing an important role. The extent to which robotics can benefit from areas such as machine learning, cognitive modeling, etc leaves us with optimism, where we can hope for the newest robots to be cognitively much more advanced than their precursors.

References

[1] A Brief History of Robotics. `http://robotics.megagiant.com/history.html`.

[2] Chemotaxis. `http://en.wikipedia.org/wiki/Chemotaxis`.

[3] Fixed action pattern. `http://en.wikipedia.org/wiki/Fixed\_action\_pattern`.

[4] Movement: Taxis and Kinesis. `http://www.sparknotes.com/biology/animalbehavior/orientationandnavigation/section1.html`.

[5] Shakey. `http://www.ai.sri.com/shakey/`.

[6] The Frame Problem. `http://plato.stanford.edu/entries/frame-problem/`.

[7] William Grey Walter. `http://en.wikipedia.org/wiki/William_Grey_Walter`.

[8] J. S. Albus. The NIST Real-time Control System (RCS): an approach to intelligent systems research. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:157–174, Apr. 1997.

[9] R. C. Arkin. *Behavior-based Robotics*. The MIT Press Cambridge, Massachusetts, 1998.

[10] J. Borenstein and Y. Koren. Potential field methods and their inherent limitations for mobile robot navigation. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 1398–1404, 1991.

[11] R. A. Brooks. Intelligence without representation. *Artificial Intelligence*, 47:139–159, 1991.

[12] C. Chengtao, Y. Yongjie, and Z. Qidan. Hybrid Control Architecture of Mobile Robot Based on Subsumption Architecture. In *Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation*, pages 2168–2172, Luoyang, Henan, June 2006.

[13] H. J. Chiel, L. S. Sterling, and R. D. Beer. A biological perspective on autonomous agent design. In *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*. The MIT Press Cambridge, Massachusetts, 1990.

[14] X. Feng and W. Li. Behavior fusion for robot navigation in uncertain environments using fuzzy logic. In *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1790–1796, 1994.

[15] J. G. Greeno. Gibson's Affordances. *Psychological Review -New York-*, 101:336, 1994.

[16] J. Lygeros, S. Sastry, M. Egerstedt, and K. Johansson. Behavior Based Robotics Using Regularized Hybrid Automata. In *Proceedings of the 38th IEEE Conference on Decision and Control*, volume 4, pages 3400–3405, 1999.

[17] A. A. Masoud. Using hybrid vector-harmonic potential fields for multi-robot,multi-target navigation in a stationary environment. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 3564–3571, 1996.

[18] A. Meystel. Planning in a hierarchical nested controller for autonomous robots. *25th IEEE Conference on Decision and Control*, 25:1237–1249, Dec. 1986.

[19] R. R. Murphy. *Introduction to AI Robotics*. The MIT Press Cambridge, Massachusetts, 2000.

[20] S. Murray. Reinventing Shakey. *Workshop on Logic-Based Artificial Intelligence*, 1999.

[21] G. Pezzulo. Schemas and Schema-based Architectures, 2007. `http://www.eucognition.org/wiki/index.php?title=Schemas\_and\_Schema\-based\_Architectures`.

[22] P. Pirjanian. Behavior coordination mechanisms – state-of-the-art. CMPSCI Tech. Rep. IRIS-99-375, Institute of Robotics and Intelligent Systems, School of Engineering, University of Southern California, 1999.

[23] R. Stenzel. A behavior-based control architecture. In *Proceedings of the 2000 IEEE International Conference on Systems, Man, and Cybernetics*, volume 5, pages 3235–3240, 2000.

[24] B. Strunz, C. Flanagan, and D. Toal. Subsumption Control of a Mobile Robot. In *Proceedings Polymodel 16, Applications of Artificial Intelligence, Sunderland*, pages 150–158, 1995.

[25] T. E. Whalen, A. Cornell, H. J. W. Spoelder, P. Rusu, and E. M. Petriu. Behavior-based neuro-fuzzy controller for mobile robot navigation. In *IEEE Transactions on Instrumentation and Measurement*, volume 52, pages 1335–1340, 2003.