The Use of State in Intelligent Control

Joanna Bryson
Intelligent Systems Laboratory
Edinburgh University
Level 8, Appleton Tower
Crichton Street
Edinburgh EH8 9JZ
UK
phone: [44] (131) 650 4354
fax: [44] (131) 650 3461
`joanna@ai.mit.edu`

February 8, 1996

Abstract

Much of the success of behavior based robotics can be attributed to the minimal representation approach proposed by Brooks in the mid-eighties(Brooks 1991). However, this has lead to a general "fear of state" which in turn leads to a neglect of the role of state in intelligent control. This paper discusses the need for propagating decision points in action selection in order to coordinate complex behavior. It also proposes a mechanism for doing this while maintaining reactive responsiveness to the real world. This mechanism involves the use of fixed action patterns and specialized behavior prioritization. The result is to focus control attention without prescribing a fixed course of action.

Topic Areas: Internal world models, Action selection

1 Introduction

One of the key components of the successes of the "new AI" approach to autonomous agents has been the abandonment of the practice of constructing and reasoning from explicit, internal models of the environment. In "Intelligence Without Representation" (Brooks 1991) Brooks documents a methodological approach, subsumption architecture, that led to the first generation of robots able to autonomously and robustly negotiate their worlds at animal-like speeds. The heart of this advance is the decomposition of intelligent control into units of *behavior.* Each behavior consists of a finite state automata (augmented with a simple clock) that produces actions as a fairly direct reaction to its perception of the environment. Each behavior can interfere with other behaviors only through affecting their inputs and outputs in strictly limited fashions. Brooks found in his experimentation that:

> We have reached an unexpected conclusion (C) and have a rather radical hypothesis (H)
>
> C — When we examine a very simple level intelligence we find that explicit representations and models of the world simply get in the way. It turns out to be better to use the world as its own model.
>
> H — Representation is the wrong unit of abstraction in building the bulkiest parts of intelligent systems. [(Brooks 1991) p.140]

This representational minimalist approach has proven useful not only for fairly complex robots like Polly (Horswill 1993) and Genghis (Brooks 1989), but for telephone network balancing (Appleby & Steward 1994) and laser head control (Pebody 1995).

Unfortunately, this work has been misconstrued as an extremist position that intelligent control should require *no* state, and the misconception that a reactive architecture is a stateless one. Subsumption architecture is composed of augmented finite *state* machines; state is a necessary element of keeping place within a control structure. Also, it quickly becomes apparent that a system with no representation can't learn. After all, representation could be defined as "the way memory is stored and accessed," and memory as "what changes when something learns." The ability to learn is present in even the simplest animal life; it is likely to be a necessary component of any intelligent behavior for any other autonomous system.

This paper concentrates more on the former area, which has also been neglected in the purge against state. In this paper I discuss what kinds of state are necessary for straightforward intelligent control, even without long-term skill or knowledge acquisition. My hypothesis is that without better ways to manage decision and control state it is very difficult to get complex behavior, particularly where this behavior requires the execution of sequential tasks. I begin with an examination of the relationship between control and state in well documented robotics experiments using both "traditional" model-based control and subsumption architecture. I then propose an organizational structure based on biological precedent, and describe the results of preliminary experiments in simulation using this model.

2 Robotic Memory Examined: Shakey vs. Genghis

Shakey is a robot constructed used as an experimental platform by the Artificial Intelligence Center at SRI International between 1966 and 1972 (Nilson 1984). This was one of the most extensive intelligent robotics projects ever, and many of Shakey's achievements have yet to be fully replicated. Genghis, in contrast, is an insect-like robot that's control was developed very quickly under subsumption architecture in 1989 (Brooks 1989), but could more successfully negotiate a dynamic, complex environment.

Parallels between Shakey's architecture and that of reactive systems have been drawn before in an effort to belittle the more recent accomplishments. The viewpoint of this paper is rather that the fact experimentation forced Shakey to converge towards a more reactive architecture validates the new approach. More importantly, as often happens in science, a new paradigm now allows us to better understand and exploit the older results.

2.1 Shakey and its Model

SRI's objective in creating Shakey was

> ...to develop concepts and techniques in artificial intelligence enabling an automaton to function independently in realistic environments. These concepts shall be demonstrated by means of a breadboard, mobile vehicle containing visual, tactile, and acoustic sensors, signal processing and pattern-recognition equipment, and computer programming. Primary goals shall be the solution of incompletely specified problems (requiring creation of intermediate strategies and goals) and improvement of performance with training experience. [(Nilson 1984) p.4]

Initially, Shakey's model of the world consisted of two different representation forms: a map-like grid model and an axiom model. Eventually the model was shifted to a single, axiom based model that was used for "all its operations."[p.19] The foundations of this model were simple predicate assertions about five classes of entities: doors, wall faces, rooms, objects and the robot. These predicates held simple information about their entity, such as its name or coordinate location, or somewhat more complex interdependencies like which rooms or faces a door connects, or what room an object is in.

Shakey's creators did not consider the predicate assertions to be the limits of their model; they considered the model to have more complex statements including conjunctions and disjunctions. They found they ran into difficulty maintaining these higher levels of the model when the robot performed an action. This is called "the frame problem" in AI literature. For Shakey, SRI decided that action routines should only update the most primitive predicates. All other statements were stored with the predicates on which they depended. When the system fetched a non-primitive statement, its primitives were tested to see if they were still in the model, if not the non-primitive was also removed. This fairly simple form of truth-maintenance meant that Shakey derived most of its reasoned information in real time.

2.2 How Shakey Used Its Model

Shakey used this model for three purposes:

1. to plan actions that would solve its goals (using STRIPS,)
2. to drive its actions, and
3. to coordinate its distributed control system.

The first purpose is the most obvious to people familiar with traditional artificial intelligence. STRIPS is essentially a rule-based system; the assertions of the model provide an initial database out of which plans are constructed. A plan is simply a list of operations that would move from the starting (current) state to the goal state.

In Shakey, each operation in STRIPS had an equivalent "action" in PLANEX, the system responsible for executing a generalized version of the plan. The generalization process is claimed as a feature allowing a plan once built to be reused in a variety of circumstances. However, experienced roboticists can recognize that such a system would also allow for the robust execution of a plan in an uncertain environment. The subgoals of each step of the plan, that is, the precondition/predicate each step is intended to enable, are carefully indexed so that at any point the plan can be shifted as far towards its ultimate goal as the robot's model of its current state allows.

The PLANEX actions look like regular function calls with elements of the model as arguments, but their primitives include the interface to Shakey's sensors and actuators, each of which could return success and failure values. Shakey's control system was ultimately highly hierarchical and distributed, so that returning error values through the system was deemed as too awkward. By the time a function "smart" enough to be able to cope with a particular failure received the flag, it could no longer tell at what level the error had occurred. So Shakey reported its errors simply by recording the results of its actions, successful or otherwise, in its internal model. Every "higher" function then had the responsibility of checking the model at every step to ascertain whether the previous action was successful. To this extent, Shakey used its model to simplify its control.

2.3 What Genghis Does Instead

Genghis (Brooks 1989) (and the other subsumption-architecture robots built before and after it) performed most of these same functions without any internal representation. The "most" is telling, but the functions it did perform, it largely performed better and faster.

Genghis also used its model to communicate between its distributed and hierarchical goal/action levels, but for Genghis that model wasn't an internal representation, but was the external world itself. Under subsumption architecture, behaviors are triggered directly by senses, which are continuously active. In fact, all behaviors are operating continuously and in parallel. Their outputs, which are the actions of the robot, vary appropriately with the state of the world.

A consequence of this form of action selection is what Agre and Chapman call "reactive planning." (Agre & Chapman 1988) Agre and Chapman (and Brooks) deny the utility of an extended sequence of actions based on formal reasoning from a static model of the world. The world is continuously dynamic, they argue, and the only appropriate way to plan is to select the action that *right now* seems most appropriate given the goal. This methodology both ensures that an agent opportunistically takes advantage of favorable environmental conditions that can unexpectedly advance its goal, and means there is no expensive or brittle plan revision when actions fail to have their expected consequence, whether because of failure or because of a change in the environment.

While most classic planners would usually miss these benefits of reactive planning, Shakey's PLANEX goal generalization and execution system can also be seen as reactive. It is opportunistic and avoids replanning. Of course, it reasons from its model, which could contain faulty information. If Genghis' sensors provide noisy information, the error is masked by the fact the that the system resamples its environment tens of times a second. A brief inappropriate motion can be quickly compensated for by the sense/action loops that compose behavior based systems.

The one thing Shakey does with its model that Genghis cannot do is create new plans (reactive or otherwise) in response to new goals. For Genghis, its highest-level goals are instantiated in its top control layer, which has been carefully and complexly engineered. Without the introduction of flexible internal state at this highest level, there can be only one goal.[1]

2.4 Control and Short Term Memory

Let's do a quick thought experiment on the nature of tasks that can be done with and without state. For our experiment, let's return to the most basic situated platform: a small mobile robot trying to navigate around an office without getting stuck. Let's say the robot has three sensors mounted one straight ahead, one angling to the right of forward and one angling to the left of forward. Each sensor returns a value for a distance to the nearest obstacle within the response cone radiating in front of it, and each response cone is directly adjacent to its neighbor so that there can be no hidden obstacle between the center sensor and one of the two flanking ones. In this case, a robot could continuously choose to run in the direction of whichever sensor reports the longest clear path.[2] This robot needs no state for this navigation. There can be constant loops that introduce rotation into the trajectory of the robot if the clearest path is to one side or another.

Now imagine that this same robot has forward bump sensors. Instead of continuous values, it only receives a signal when it has already come too close to an obstacle. In this case, the robot needs to back up and turn away from the obstacle to avoid it. Notice that

[1] Flexible state in a top level goal was introduced for a subsumption-architecture robot named Toto (Mataric 1990). This robot *could* take multiple goals — it could be asked to navigate to any of the locations it had learned.

[2] This is essentially the obstacle avoidance algorithm for Polly (Horswill 1993), though the sensors are three subdivisions of the visual scene returned by a single camera.

during this avoidance, the robot is not getting any signal from its sensors. In order to complete the entire maneuver, the robot must *remember* not only that it hit its bumper, but where it is in the sequence of motions to compensate. In fact, retaining the initial trigger of the action pattern is probably not important, but keeping track of the progress of the behavior sequence until it terminates is.

Genghis actually does have touch sensors. They trigger behaviors with timers in them that dominate the "default" behaviors until the timers have run out. (These timers are the reason subsumption architecture behaviors are defined as *augmented* finite state machines.) But the point of the thought experiment is that it answers the question "When do you need control state?" *Whenever a time-bounded event should influence behavior after the end of that time boundary.*

Even this rudimentary change of state is called "learning" by some researchers. But by conventional use, we would only call such a change "learning" if it leads to permanent changes in behavior; that is, if it changes the state of long term memory. If the event only affects immediate behavior, then if its trigger is external we call that behavior a "reaction." If the trigger was *internal,* such as an arbitrary choice among possible goals to pursue, we might call it a "decision."[3] Either way, this information is stored in short term control memory.

2.5 "Short Term Memory" in Ghengis and Shakey

With our new understanding of "reactions" and immediate behavior, we can revisit our thought experiment about the two kinds of sensors on a mobile robot. The robot's control under the continuous distance sensors could also be understood in terms of short-term memory and state. The difference is only that here the time to sample the environment and execute the reaction is extremely brief, so recovery from error is so quick and fluid as to be transparent. Thus in both cases, the short term memory necessary for reacting appropriately to the environment is buried in the actual behaviors that process the sensor input. [4]

Let us now return to Shakey — to what extent did it really need its model? The problem with Shakey was that the cost of achieving new sensor input was very high; it took several minutes to take and analyze a picture. Therefore it was much more reasonable for Shakey to make as much progress as possible on remembered state. Clearly, if Shakey had been able to recognize the salient features of its environment with the fast, efficient perceptual routines that have been the hallmark of behavior based robotics, its navigational trajectories and motion could have appeared more smooth, fast and intelligent. But if it had no model at all, the project objective of being able to communicate goals or tasks to it, particularly tasks involving objects in other rooms, would have been nearly impossible. Also, Shakey would have lost its ability to create new, task-specific orderings of its behaviors into general/reactive

[3]Notice that an external observer might not be able to discern the difference between a "reaction" and a "decision."

[4]Cognitive neuroscience indicates that short term memory for animals is also contained in the perceptual processing systems of the appropriate sensor modalities. Long term memory is a separate and parallel process involving completely different regions of the brain.

plans.

3 A Better Way to Maintain Decisions

The previous section establishes the need to maintain decision state in control in order to execute complex behavior, or at least any behavior that requires actions *because* of an event *after* the event. For a conventional planning system, like Shakey, there are two locations for decision state. One is the plan itself, the other is the goal. Essentially, a plan is an ordered set of behaviors that have been previously determined to be appropriate, given a goal and a world state. Choosing a new goal (or subgoal) also preserves state about the robot's intentions. In Shakey, maintaining the goal separately from the plan allowed the robot to compute a new plan if the current one failed. This in theory allows for robust, coherent behavior. However, robots running planning systems still tend to have too rigid and brittle a behavior, as well as spending too much time in computation. In the first part of this section, we propose a new method of organizing control state which is ethologically inspired and at least partially reactive.

Recall that Genghis also has control state for decisions as provided under subsumption architecture. But this is a simple timer used to inhibit other behaviors while a particular unit "takes control." In other words, this is an exception handler, built into subsumption architecture because of a particular exception (Herbert's loosing visual contact with a can it was trying to pick up, see (Connell 1990)). But in animals, decisions are not occasional events. The overall behavior of a creature at any given moment is a choice of context, a focusing on a particular set of capabilities. For an example, consider rat navigation. There is now strong neurophysiological evidence that rats do not constantly maintain a single spatial representation global to their experience, but rather have many spatial matrices which are drawn into "working memory" when appropriate, indexed by context (McNaughton, Barnes, Gerrard, Gothard, Jung, Knierim, Kudrimoti, Qin, Skaggs, Suster & Weaver 1996). Therefore, at every instant a rat's motions are to some extent guided by a fixed decision point, in this case recognition of location and task. The second part of this section reports experiments which indicate that such decisions may be used to focus limited attention resources and thus enable behavior that is too complex for flat architectures.

3.1 The Model

The model I propose provides two structural mechanisms for maintaining decision state outwith an individual behavior. The first is the *Action Pattern* (AP) which is based on the concept of Fixed Action Patterns familiar from ethology. Action patterns are simple sequences of actions which once started, can only be interrupted from internal failure. In nature, FAPs serve to seriously reduce the combinatorial complexity of action selection. In my model, APs can be composed of traditional subsumption architecture behaviors, action primitives, or other APs. They can also, like ordinary behaviors, be linked to sensor inputs which can essentially inhibit their availability.

The second structure is a *Competence*, which is a group of action patterns associated with a goal. When an internal decision point is reached to pursue a particular goal, the control mechanism turns its attention to a prioritized set of action patterns associated with it. Of all the APs that currently have their sensor preconditions met, the highest priority one is triggered. Priority can also be understood in terms of activation level; when a goal's level is raised to its threshold, it excites appropriately the associated behaviors. Whenever an active AP fires, it also "habituates" by lowering its priority level. This guarantees termination of a competence, and also that in the case of multiple, equally weighted and equivalently prepared strategies, different ones will be tried. If a competence needs to repeatedly execute a particular action, it can be retriggered and pick up where it left off. This allows for a level of reactiveness, such as is exhibited when an animal periodically breaks from feeding to survey its environment.

The competence itself is also reactive, another instantiation of the concept of a reactive plan. The competence doesn't predict deterministically the next action pattern to be called, or "expect" that an action pattern will "succeed". Every AP is selected based on the current environment. The action patterns themselves are less reactive, although they can terminate prematurely either due to sensor checks or radical failure of a constituent action. This is in keeping with the current ethological definition of a FAP.

The intent behind the design of this architecture is to provide a grammar for expressing animal behavior, not only instinctive but learned. Thus the concern is to present both a straight-forward way for researchers to develop a lexicon of provided knowledge for a robot, and mechanisms for the learning of new behavior. In animals, acquisition of complex behaviors seems to come from imitation of conspecifics, recombinations of old knowledge, and in relatively rare occasions (mostly in humans) developing of new skills through long periods of training. This architecture was developed particularly with programmed knowledge and learning by recombination in mind; so far research has only been carried out in the former area.

3.2 An Implementation

I will now describe some preliminary experimentation with this model, performed in simulation. As indicated above, these experiments were only to ascertain the plausibility of such a control model, and its programmability. However, partly because of our[5] interest in developing a learning system, we chose to use the action primitives, sensing capabilities, domain and tasks described in Steve Whitehead's well-known, blocks-world based learning thesis, "Reinforcement Learning for the Adaptive Control of Perception and Action" (Whitehead 1992). Another motive for this choice of domain was that we have a functional visual routine processor of the type postulated in Whitehead's work (see (Horswill 1995)), and so have hope of implementing the system on a real robot. Finally, Chris Malcolm, the head of Behavior Based Assembly Robotics at the University of Edinburgh has asserted that "no behavior-based robot can ever do assembly, because it's too complicated."(Malcolm 1992) His group advocates hybrid systems of traditional planners with behavior based systems simplifying

[5]These trials were begun with the advice of Ian Horswill, now of Northwestern University.

the actual manipulations in the real world.

As in subsumption architecture, the competences of this model have to be carefully engineered. To succeed the action patterns must be prioritized in such a way that they will converge to the goal if possible. The priority levels are arbitrary except that, where desirable, they discriminate APs that could be active (sensor preconditions are met) at the same time. Figure 1 shows a competence that achieves the end result of Whitehead's thesis: it grasps a green block, even if that block is buried under an arbitrary number of other blocks[6]. As in Whitehead's thesis, no attempt is made to optimize the block that is first selected for attention.

Weight	AP Name	AP Actions
100	pick-up-green	ACTN-in-hand = t, ACTN-color = green
50	lose-block	object-in-hand = t move-ACTN-to-table place-object-at-ACTN
50	unbury-green	object-in-hand = nil, ATTN-color = green, markers-vertically-aligned = t move-ACTN-to-stack-top grasp-object-at-ACTN
40	synch-on-green	ATTN-color = green move-ACTN-to-ATTN
20	find-green	move-ATTN-to-green

Figure 1: pick-up-green: A simple competence and the action patterns that compose it. The weights are part of the competence, not the AP.

As can be seen from this code, Whitehead's thesis was concerned mostly with the perceptual task. The perceptual operations and state variables[7] derive from David Chapman's implementation of Shimon Ullman's "Visual Routines Theory" of the visual system (Chapman 1990). Obviously, in a real robot, grasping and placing would also have to be competences. Perceptual learning is also a real, ethological issue: current developmental research indicates that failures of categorization often result from a very real failure to *perceive* difference, and require the development of new skills (McGonigle & Chalmers in press).

Ullman's VRP theory is based partially on psychophysiology studies that indicate we have a limited number of units of visual attention we can fix to various objects in our surroundings. Ullman postulates that we can compare features of these markers, for example their color and relative location. Given these limited resources, I found it very difficult to implement the

[6]Actually, since Whitehead's controller has a 2-bit stack-height sensing variable, he presumably allows only 3 covering blocks and does not allow restacking.

[7]What I call "markers", Whitehead calls "frames." Due to my AI dialect, I chose Chapman's terminology.

task Whitehead stated as his original goal, putting a red block on a green block, except as two separate competences. By focusing *behavioral* attention I was able to optimize *perceptual* attention limits. Since the behavioral attention can be explained with a simple neuron-based model of activation levels, but a marker requires association with complex operations, this is a reasonable tradeoff.

Green-on-Green			**Copy-Demo**	
Weight	AP Name		Weight	AP Name
100	green-on-green		100	copy-demo
80	place-first-green		80	check-next-goal
80	place-green-on-green		60	add-blue
40	check-home		60	add-green
40	get-another-green		60	add-red
40	get-first-green		60	place-block
30	clear-home		50	lose-block
30	lose-block		40	clean-home
			20	start-over
			Synch-Action-Up	
			Weight	AP Name
			100	synch-action-up
			50	look-at-home-again
			40	home-act-up

Figure 2: green-on-green and copy-demo: Both tasks also required the competence pick-up-green shown above; copy-demo has pick-up-red and pick-up-blue as well.

For green-on-green[8] (see figure 2) I did add an additional marker, “home”, for the location of the produced stack. I presumed in a real animat this could be done by association with privileged body orientation or other environmental cues. This marker has considerably less operations than “action” or “attention”; it is more like the location “hand” in Whitehead's thesis than a true marker. I also implemented a trivial version of “the copy demo,” (Winston 1972) a block stacking task where the goal (another location) is another stack of blocks, also viewed. With the goal thus externalized I was able to simplify the structural competence, although I found it necessary to create a specialized perceptual competence for comparing the piles. This task, the only one implemented with a fully debugged controller, took less than two hours to program, debug and test in simulation.

4 Conclusions and Implications

This paper has reviewed the role of state in robotics, with particular attention to the need for propagating decisions through the control structure in order to carry out complex, se-

[8] One of the first learning procedures implemented will be the obvious generalization procedure for a competence, where sensory difference in a goal is “variablized” and repeated throughout the competence.

quential tasks. I have presented a model whereby such propagation can be achieved via the ethologically inspired mechanisms of fixed action patterns and graded increases of activation levels. I have also described some preliminary experiments in implementing this model. Clearly the model needs to be refined, but it serves as a demonstration of the main argument of the paper. It should also be acknowledged that, as an architecture, it is not particularly unique. It resembles the architectures presented in (Correia & ao 1995), which is also FAP inspired, and the competences resemble TR (Nilsson 1994). Both these architectures have been successfully implemented in mobile robots. But both projects have substantially different conceptualizations of their underlying model from the one presented here.

Understanding the relationship between control and state may bring us closer to understanding the notion of control *as* state. Possibly an appropriate representation of motor control is an integral part of an appropriate representation of memory for motor tasks. The idea of associating goal tasks with motor memories comes close to the model of mouse navigation researched by McNaughton (McNaughton et al. 1996). They have a body of experimental evidence indicating that mouse spatial representation consists of matrices of ideothetic information combined with loose associations to goals and landmarks. These landmarks serve both to index and to recalibrate the ideothetic information. Perhaps if we can understand the minimal state needed to control an agent's motions, then we will know the minimal state needed to record them.

Acknowledgements

Leslie Kaelbling first recognized the similarity between my work and Shakey, and loaned me her copy of the original TR. Ian Horswill, Gill Pratt and Brendan McGonigle have helped me think more clearly during various stages of this work. Limor Fried worked on implementing this work on a real robot and provided valuable feedback. Bill Smart, Phil Kime and Pushpinder Singh helped with the draft.

References

Agre, P. E. & Chapman, D. (1988), What are Plans For?, AI memo 1050, MIT, Cambridge, MA.

Appleby, S. & Steward, S. (1994), 'Mobile Software Agents for Control in Telecommnications Networks', *BT Technology Journal*.

Brooks, R. A. (1989), 'A Robot That Walks: Emergent Behavior from a Carefully Evolved Network', *Neural Computation* **1**(2), 253–262.

Brooks, R. A. (1991), 'Intelligence without Representation', *Artificial Intelligence* **47**, 139–159.

Chapman, D. (1990), Vision, Instruction, and Action, Technical Report 1204, Massachusetts Institute of Technology Artificial Intelligence Laboratory, Cambridge, Massachusetts.

Connell, J. H. (1990), *Minimalist Mobile Robotics: A Colony-style Architecture for a Mobile Robot*, Academic Press, Cambridge, MA. also MIT TR-1151.

Correia, L. & ao, A. S.-G. (1995), A Useful Autonomous Vehicle With a Hierarchical Behavior Control, *in* F. Moran, A. Moreno, J. Merelo & P. Chacon, eds, 'Advances in Artificial Life (Third European Conference on Artificial Life)', Springer-Verlag, Berlin, Germany, pp. 625–639.

Horswill, I. D. (1993), Specialization of Perceptual Processes, PhD thesis, MIT, Department of EECS, Cambridge, MA.

Horswill, I. D. (1995), Visual routines and visual search, *in* 'Proceedings of the 14th International Joint Conference on Artificial Intelligence', Montreal.

Malcolm, C. (1992), Personal communication.

Mataric, M. J. (1990), A Distributed Model for Mobile Robot Environment-Learning and Navigation, AI memo 1228, MIT, Cambridge, MA.

McGonigle, B. & Chalmers, M. (in press), *The Growth of Intelligence in Complex Systems*, MIT Press, Cambridge, MA.

McNaughton, B., Barnes, C., Gerrard, J., Gothard, K., Jung, M., Knierim, J., Kudrimoti, H., Qin, Y., Skaggs, W., Suster, M. & Weaver, K. (1996), 'Deciphering the hippocampal polyglot: the hippocampus as a path integration system', *The Journal of Experimental Biology* **199**(1), 173–185.

Nilson, N. (1984), Shakey the Robot, Technical note 323, SRI International, Menlo Park, California.

Nilsson, N. J. (1994), 'Teleo-Reactive Programs for Agent Control', *The Journal of Artificial Intelligence Research* **1**(1), 139–158.

Pebody, M. (1995), Learning and Adaptivity: Enhancing Reactive Behaviour Architectures in Real-World Interaction Systems, *in* F. Moran, A. Moreno, J. Merelo & P. Chacon, eds, 'Advances in Artificial Life (Third European Conference on Artificial Life)', Springer-Verlag, Berlin, Germany, pp. 679–690.

Whitehead, S. D. (1992), Reinforcement Learning for the Adaptive Control of Perception and Action, Technical Report 406, University of Rochester Computer Science, Rochester, NY. also a PhD Thesis from the same date.

Winston, P. H. (1972), The MIT Robot, *in* B. Meltzer & D. Michie, eds, 'Machine Intelligence, 7', John Wiley and Sons, New York, New York, pp. 431–463.