In this chapter, we introduce the concepts that we need later on when exploring the various approaches. Moreover, we need such a framework if we actually want to build agents. One important concept that we discuss is that of the complete agent. Complete agents are inspired by natural agents, animals and humans, which are—quite obviously—capable of surviving in the real world. They are ''complete'' because they incorporate everything required to perform actual behavior. (Standard computer programs, for example, are not complete because they cannot behave in the real world.) We argue that it is such complete agents that we want study and synthesize. We provide a characterization of what we mean by complete agents, and we show that if we want to model, to synthesize such agents, we must take into account some special considerations relating to the idea of emergence, that is, to the fact that behavior emerges from the agent-environment interaction. Emergence is in turn a consequence of the frame-of-reference problem, which conceptualizes the relationships among those involved in the design process, namely the designer (who is often also the observer), the natural agent (if we are doing modeling work), the agent to be designed, and the environment. One important implication of frame-of-reference considerations is that behavior cannot be reduced to an internal mechanism. This in turn necessitates a new design methodology, which is this chapter's central topic.

We begin the chapter with a characterization of complete agents and discuss a number of basic concepts like adaptivity, autonomy, self-sufficiency, embodiment, and situatedness. We then turn to agents—both simulated and real robots—and discuss how they can be used as modeling tools. We examine the pros and cons of working with real robots and with agent simulations. We also compare this new kind of agent simulation with more traditional forms of simulation. We then outline the framework for design that focuses on emergence, including a description of the frame-of-reference problem. Finally, we discuss what we mean by a good explanation

and how we can find explanations of agent behavior by running experiments.

This chapter is difficult and covers a lot of ground. This is unavoidable. At first reading, all the points may not become immediately clear. All the issues raised here, however, will be illustrated in greater detail later on. The reader may find it helpful to return to this chapter after having read through some of the subsequent chapters.

4.1 Complete Autonomous Agents

Biological agents have to perform a number of tasks: searching for food, eating and drinking, grooming, reproducing, and caring for their offspring. The term ''task'' is normally used in a design context to designate something the agent needs to get done. Typical tasks for autonomous robots, for example, are marking all the mines in a mine field with color, or mowing the lawn of a soccer field. Note that the task of mowing the lawn implies certain desired behaviors on the part of the agent. What is really meant is that the agent's task is to keep the grass short. And because the designer can't think of any other way to accomplish the job, he simply equates the task with the method, that is, with the behavior by which the task is to be achieved, namely mowing. Note that animals don't have tasks. Rather, a task is an observer-based attribution summarizing the effect of certain behaviors of the animals. In the field of embodied cognitive science, researchers often talk about tasks of animals. What they mean is either the behavior involved—collecting food—itself or the effect of the behavior, that is, the fact that if the animals behave in a particular way, the food ends up in the nest. What is important is that we observe the frame-of-reference problem: There need be no internal representation of the task within the agent. Often, the distinction is not so relevant: Both task and desired behaviors can be used to specify what an agent should do.

The ability to survive in complex environments is a given for all biological systems. Achieving this ability in artificial agents turns out to be an extremely hard problem. Complete autonomous agents are physical systems that are able to resolve these issues. For fun and for historical reasons we also call these complete autonomous systems ''Fungus Eaters.'' Let us briefly look at the story of these

''Fungus Eaters.'' They illustrate the main intuitions underlying the embodied cognitive science framework.

In 1961 the Japanese psychologist Masanao Toda[1] proposed to study ''Fungus Eaters'' as an alternative to the traditional methods of academic psychology (Toda 1982, chap. 7). Rather than performing ever more restricted and well-controlled experiments on isolated faculties (memory, language, learning, perception, emotion, etc.) and narrow tasks (memorizing lists of nonsense syllables, letter perception on degraded stimuli, etc.), we should study ''complete'' systems, though perhaps simple ones. ''Complete'' in this context means that the systems are capable of behaving autonomously in an environment without a human intermediary. Such systems have to incorporate capabilities for classification, for navigation, for object manipulation, and for deciding what to do. The integration of these competences into a system capable of behaving on its own, according to Toda's argument, will yield more insights into the nature of intelligence than looking at fragments of the complex human mind.

The ''Solitary Fungus Eater'' is a creature—in our terminology, an autonomous agent—sent to a distant planet to collect uranium ore (see figure 4.1). The more ore it collects, the more reward it will get. If feeds on a certain type of fungus that grows on this planet. The ''Fungus Eater'' has a fungus store, means of locomotion (e.g., legs or wheels), and means for decision making (a brain) and collection (e.g., arms). Any kind of activity, including thinking, requires energy, if the level of fungus in its fungus store drops to zero, the Fungus Eater dies. The Fungus Eater is also equipped with sensors, one for vision and one for detecting uranium ore (e.g., a Geiger counter).

The scenario Toda describes is interesting in a number of respects. Fungus Eaters must be autonomous: They are simply too far away to be controlled remotely. This autonomy in turn implies situatedness: Because they cannot be remote controlled, they have to view the world from their own perspective; that is, the only information the agent has available is acquired through the sensors in interaction with the environment. Fungus Eaters must be self-sufficient, because there are no humans to exchange their batteries and to repair them. They must be embodied, otherwise they would not be able to collect anything in the first place. All this implies

[1] This is our own interpretation of his paper; Toda may not agree with it.

Figure 4.1 Toda's Fungus Eater, a complete autonomous agent. The robot is operating on a distant planet. Its task is to collect uranium ore. It feeds on a certain type of fungus. It is autonomous (too far away for remote control), self-sufficient (it must take care of its own energy supply which, in this case, is a particular type of fungus that grows on this planet, thus the name Fungus Eater), embodied (it exists as a physical system), and situated (its knowledge about the environment is acquired through its own sensory system). In the figure, it is in the process of devouring fungus.

that they must be adaptive, because the territory in which they have to function is largely unknown. These concepts are fundamental to embodied cognitive science, and we now discuss each in turn.

Before we do so, however, let us first examine another reason why Fungus Eaters are of particular interest for the study of intelligence, one that relates to evolutionary considerations. Nature has always produced Fungus Eaters, that is, creatures capable of surviving in the real world. There are, for example, the single-cell entities that emerged from the primordial soup 3.5 billion years ago. Only 550 million years ago, the first fish and vertebrates arrived, insects 450 million years ago. Reptiles came 370 million years ago, dinosaurs 330, and mammals 250 million years ago. Pri-

mates appeared 120 million years ago, the great apes 18 million years ago, man in its present form only 2.5 million years ago. Writing was invented less than 5,000 years ago. Based on these considerations, Brooks (1991a) argues that the really hard part for nature was to get to the level where creatures could move around and had sensory abilities. Once that was in place, things became much simpler. If we do not understand this sensory-motor basis, we have no chance of ever understanding intelligence. This is another fundamental reason why we must study Fungus Eaters, that is, complete autonomous systems.

Self-Sufficiency

MULTIPLE TASKS AND BEHAVIORS

Self-sufficiency means an agent's ability to sustain itself over extended periods of time. This implies that the agent must maintain its energy supply. A biological agent must eat and drink. Moreover, it has to eat and drink the right combination of foods. A prerequisite of eating and drinking is that the food and drink be there: Humans have to go to the grocery store or a restaurant; an animal typically has to look for food in the environment, an activity called foraging. An agent must also take care of itself; that is, it has to stay sufficiently clean, and it has to try not to get hurt. In other words, it also has to avoid predators. Moreover, it has to get enough sleep. If these conditions are fulfilled, the biological agent can engage in activities leading to reproduction. (Note that this description in terms of tasks is our description as observers. It has nothing to do with what is going on inside the animal.)

Similar considerations apply to artificial systems. A robot, for instance, has to maintain its battery level, or if it is fuel driven, it has to maintain a sufficient fuel supply. To be considered self-sufficient, the robot should be able to maintain its energy supply without external human intervention. Thus, a robot running off a power cable is not self-sufficient. A robot should also maintain a certain operating temperature. If it gets too hot or too cold, it might be damaged. Moreover, it should not bump into things, and it should avoid perils. In addition, robots are always designed for a particular task, or several tasks. They have to clean a factory floor, vacuum a carpet, mow a lawn, deliver mail in an office, collect soda cans, give tours of a university institute, and so on. Hence, agents

in the real world, be they animals or robots, always have to engage in multiple behaviors. From an observer's perspective, we can say that they are able to perform multiple tasks.

TRADE-OFFS AND DEFICITS

In the real world, there are always trade-offs. If a robot is collecting soda cans or food or cleaning a park, it always expends energy. So at some point, it must replenish its energy resources; that is, it must go to the charging station and plug itself into an outlet. While doing that, it cannot collect soda cans: It must remain at the charging station until its energy supply is sufficiently high again. So there is a trade-off: Doing one thing implies not being able to do another.

Note that losing energy while collecting soda cans or mowing a lawn is a given, determined by the physics of the agent: It will happen without the agent's knowing about it. If a cleaning robot is recharging, the office space gets cluttered with soda cans or the grass keeps growing without the robot's doing anything about it: Remember, the real world has its own dynamics. If it remains at the charging station for a long time, enough soda cans might have accumulated so that it is no longer possible for the robot ever to collect all of them again. Or, to put it differently, it has incurred an irrecoverable deficit. Another way of defining self-sufficiency, then, is as follows: An agent is self-sufficient if it can avoid irrecoverable deficits. In nature, evolution has ''solved'' this problem, but robot designers must explicitly deal with it. Figure 4.2 shows a robot that has incurred an irrecoverable deficit.

CIRCADIAN CYCLES

Natural environments have circadian cycles: environmental conditions that change over one day, such as lighting conditions, temperature, or humidity. Similarly artificial environments often have cycles: day-night cycles, or cycles in the frequency of people attending a place (coffee rooms are attended more during day time than at night), and so forth. Conditions for certain types of tasks are usually better during one segment of the cycle than during another. For example, an agent equipped with vision is better off during the day, whereas one with infrared (IR) sensors is better off at night, for the following reason. IR sensors are active sensors: They send out an IR signal and measure the intensity of the reflected IR light, a process that works well in the dark. By contrast, a robot equipped only with IR sensors has trouble during the day. Daylight contains

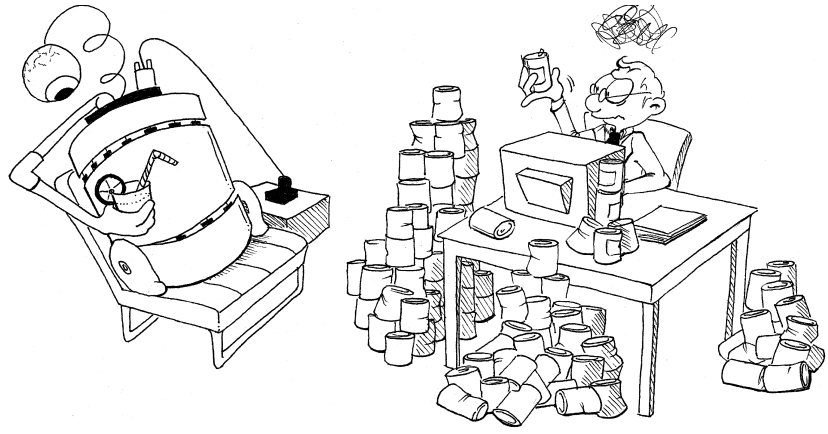


Figure 4.2 Robot incurring an irrecoverable deficit. Because the robot has been sitting at the charging station for too long, the soda cans have piled up in the meantime to a level where the robot is no longer capable of removing them all, even if it were to spend all of its "spare time," that is, all of the time it has available when not at the charging station, on can collecting. This robot is not self-sufficient.

a certain amount of IR light, which may cause interference with the reflected IR light. For the robot in figure 4.2, soda cans typically accumulate more quickly during the day. The target for a self-sufficient agent is always based on a circadian cycle: It should not incur a deficit over one cycle. If it does, then the deficit is likely to increase indefinitely, because the following day will typically bring an additional deficit. The concept of circadian cycles has not been widely used in embodied cognitive science and will not be further elaborated.

THE PROBLEM OF BEHAVIOR CONTROL

Complete systems always have several behaviors in which they must engage. Some of the behaviors will be compatible, others mutually exclusive. Because not all behaviors are compatible, a decision must be made as to which behaviors to engage in at each point in time. This is the problem of *behavior control*.

The most straightforward solution to this problem is to assume that there is an internal module or representation for each observed behavior category. For example, if we observe that a rat (or a robot) is following a wall, we might postulate that it has an internal module or a representation for wall following. Such a representation is often called an action. Because there are always multiple actions an agent has to engage in, to control behavior under this assumption,

you need a mechanism for deciding which action to choose for execution at any given point in time, that is, which internal module to excute. In other words, you have to solve the *action selection problem*.

The problem with this approach to behavior control is that the assumption of a straightforward, one-to-one mapping from a specific behavior to a specific internal action does not reflect what actually occurs in natural systems. (Even the concept of an internal action represents an assumption.) To illustrate this point, let us look at an example. Assume that you are sitting in the cafeteria talking to a friend. Your friend has to attend a class and you are trying to describe his behavior. He gets up and starts moving toward the exit, avoiding chairs, tables, and people who stand around. To describe his behavior, you may want to use terms like ''avoiding a chair,'' ''going toward the exit,'' or ''going to class,'' implying that you somehow carve up your friend's behavior into distinct segments. There are two issues of which to be aware: First, the segmentation of an agent's behavior is observer-based and largely arbitrary. For example, you could also choose a more fine-grained segmentation such as ''getting up from chair,'' ''moving left leg forward,'' ''moving right leg forward,'' and so forth. Not surprisingly, segmentation of behavior is a notorious problem in psychology and ethology. For empirical purposes such a segmentation obviously has to be made, but we need then to make explicit that we are talking about purely observer-based categories. Second, it is not appropriate to conclude that for each of these behavioral segments there is an internal module.

There are mechanisms for behavior control, however, that do not require the existence of internal actions. Chapter 6 discusses an example, Braitenberg vehicles. In fact, we think that the problem of behavior control should be approached differently than described above. This follows from one of our design principles, the principle of loosely coupled, parallel processes (see chapters 10 and 11).

Autonomy and Situatedness

We have been using terms like ''autonomous agents'' and ''autonomous mobile robots.'' In this context, autonomy generally means freedom from external control. Autonomy is not an all-or-nothing issue, but a matter of degree. Complete, total autonomy does not exist; no agent is totally autonomous. It always depends to some

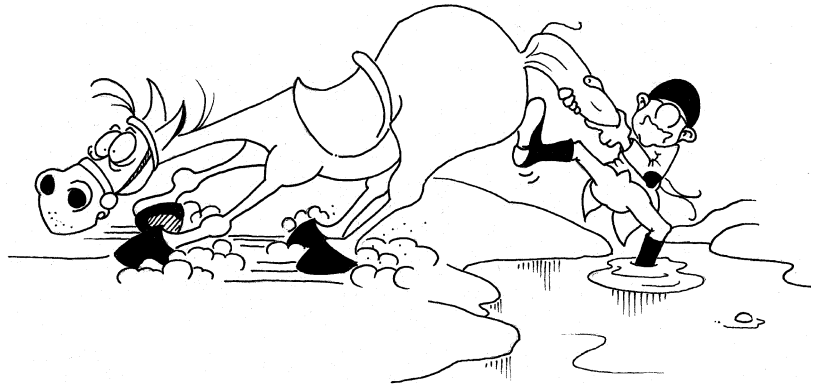


Figure 4.3 A horseback rider trying to control his horse. He is trying to force his horse to drink, not very successfully. The rider does exert some influence on the horse, and the horse is dependent on the rider for some things, but the horse is also to some degree autonomous. This is why the adage that ''you can lead a horse to the water but you can't make him drink'' has the ring of truth to it.

degree on external factors, factors beyond the agent's control. There are two aspects of autonomy here: dependence on the environment and dependence on other agents. Organisms depend on the environment for food, drink, oxygen, building materials, and the like. If agents are not capable of acquiring these resources on their own, they depend on other agents—they are less autonomous.

The main difference between dependence on the environment and dependence on other agents is that we do not attribute intentions to an environment, whereas an agent may want another agent to do certain things. Most parents want their children to do their homework and to perform well in school. We know, however, that parents have only a limited influence on their children: The latter have some degree of autonomy. The same holds for animals. We can get horses to do certain things we want them to do. But as the saying goes, ''You can lead a horse to the water but you can't make him drink,'' again implying that the horse does have a certain degree of autonomy. So, in general, agents can be influenced, and they depend on others, but they are not completely controllable, as figure 4.3 illustrates.

From this discussion it becomes clear that when we use the term ''autonomous agent,'' we mean an agent that has a certain degree of autonomy. It is not the case that an agent is either fully autonomous or not at all. From our discussion of self-sufficiency, it should be evident that self-sufficiency increases an agent's degree of auton-

omy, because a self-sufficient agent does not depend on another agent for its energy supply. The extent to which one agent can control another depends on the controlling agent's knowledge of the state and the internal mechanism of the agent to be controlled. The more precisely parents know what their children feel and think, the better they can influence them toward desired behaviors. One important reason that humans have only a very limited degree of controllability is that they have their own history, which is not, or is only indirectly and to a very limited extent, accessible to others.

Controllability and the capability of acquiring one's own history are correlated: The more an agent can have its own history, the less controllable it will be. The less parents know what their children do and what sorts of experiences they have, the less they know about what they feel and think. If they knew everything about them (including their reaction to all types of events)—which, of course, is impossible—they could easily make them do whatever they wanted, simply by manipulating the consequences of the children's actions according to what they knew the children's reactions would be. Because parents actually have only limited knowledge of their children's reactions, they have only limited control over them. Abstractly speaking, if the controlling agent (A) has access to the controlled agent's (B) internal state, and if he knows the laws by which the state of B can be influenced, A can control B completely, that is, A can get B into whatever state A wants B to be in. The less knowledge A has about B's internal state, the less A can control B. Thus, autonomy is not so much a property of an agent as a property of the relationship between agents (i.e., what one agent knows about the other). Stated differently, B has a certain amount of autonomy relative to A, and the amount of B's autonomy is—qualitatively speaking—inversely proportional to the amount of knowledge A has about B's internal state.

This property can be translated to robots. If a robot is equipped with a learning system, it can have its own experiences; that is, it can acquire its own knowledge over time. Note that this requires the agent to be situated. Recall the notion of situatedness from chapter 3: An agent is situated if it acquires information about its environment only through its sensors in interaction with the environment. A situated agent interacts with the world on its own, without an intervening human. It has the potential to acquire its own history if it is equipped with the appropriate learning mecha-

nisms. Such an agent is potentially more autonomous than its preprogrammed, purely reactive counterpart. One implication of learning is that if the agent, after learning, encounters the same situation it has previously encountered, it will react differently than earlier on. Thus the more the agent has learned in the meantime, the more experiences of its own it has had, the less it will do the same as before, and thus, the less another agent will be able to control it, because its internal state will have changed, and the second agent will now have less knowledge of its internal state than it did previously. From this we can conclude that if we are interested in building autonomous agents, we must design them with learning components, because the capacity to learn increases an agent's autonomy. An agent's degree of autonomy can, in principle, be further increased by applying evolutionary methods (described in chapter 8). If he designs a robot not directly but via an additional evolutionary process, the designer has less control over how the robot will work and how it will behave in a particular situation. Applying evolutionary techniques often makes it difficult for designers—and for other agents in general—to understand why the agent is doing what it is doing; as the agent evolves and acquires its own history, it is progressively more difficult for the designers to understand (and manipulate) its behavior. Evolution makes the agent more independent of designers, and therefore evolved agents have the potential for higher levels of autonomy.

Embodiment

Autonomous agents are real physical agents; in other words, they are embodied. Because we have talked so far exclusively about biological agents (humans or animals) or about robots, it has been implicit that the agents of interest have to be embodied. Embodiment has proven to be an essential characteristic whose importance can hardly be overemphasized. A fundamental consequence of embodiment is that embodied agents must interact with their environments. To understand this interaction, we have to study, for example, how organisms acquire experience: knowledge about the environment obtained by interacting with it. This is one of the hardest problems in the study of intelligence. The vast research field of perception is devoted to elucidating the underlying mechanisms and processes.

Embodiment implies that the agent is continuously subjected to physical forces, to energy dissipation, to damage, in general to any

influence in the environment. On the one hand, this complicates matters considerably. On the other, this often leads to substantial simplifications, because advantage can be taken of the physics involved. It has been demonstrated, for example, that walking robots can be built that require no electronic control: They are entirely brainless machines, their actions governed totally by the laws of physics.

The focus on embodied agents often leads to surprising insights, and throughout the book, we provide examples of such insights. We discuss embodied perspectives on learning, categorization, perception, memory, and sensory-motor processing. As the name of the field indicates, embodiment is at the core of embodied cognitive science. It is one of the central constituents in Brooks's (1991a,b) approach, which he called ''embodied intelligence.'' The idea that intelligence can emerge only from embodied agents is one of the fundamental assumptions of embodied cognitive science. (For other perspectives on embodiment see, for example, Lakoff 1987 and Varela, Thompson, and Rosch 1991).

Adaptivity

CHARACTERIZATION AND DEFINITION

Adaptivity is really a consequence of self-sufficiency. If an agent is to sustain itself over extended periods of time in a continuously changing, unpredictable environment, it must be adaptive. Remember that several of the definitions of intelligence given in chapter 1 alluded, in one way or another, to the concept of adaptivity, that is, the ability to adjust oneself to the environment. Thus, adaptivity and intelligence are directly related.

By adaptation, we mean that some structure is maintained in changing environmental conditions. Ashby (1960) used the term ''homeostasis,'' meaning that certain variables, the essential variables, remain within given limits (figure 4.4). Within those limits the organism can function and stay alive. This is called the ''viability zone'' (Meyer and Guillot 1990).

KINDS OF ADAPTATION

The term ''adaptation'' has various meanings and is used in different ways by different people. In our discussion, we follow McFarland (1991):

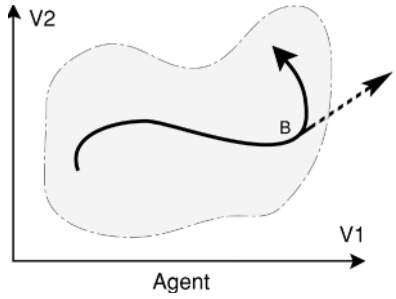


Figure 4.4 Adaptivity. The figure shows the viability zone (enclosed area) between two variables *V1* and *V2* (e.g., level of blood sugar and body fluid). Within this zone, the agent can stay alive and function. The solid arrow marks the agent's trajectory, that is, the development of the two variables over time. At point B, there is a danger that the agent might leave the viability zone (marked by the broken line) if it does not act. The agent is adaptive because it takes corrective action to prevent itself from leaving the viability zone. (Adapted from Meyer and Guillot 1991.)

Biologists usually distinguish between (1) evolutionary adaptation, which concerns the ways in which species adjust genetically to change in environmental conditions in the very long term; (2) physiological adaptation, which has to do with the physiological processes involved in the adjustment by the individual to climatic changes, changes in food quality, etc.; (3) sensory adaptation, by which the sense organs adjust to changes in the strength of the particular stimulation which they are designed to detect; and (4) adaptation by learning, which is the process by which animals are able to adjust to a wide variety of different types of environmental change.'' (p. 22)

Here are a few illustrations of the types of adaptation McFarland discusses (see also McFarland 1991):

1. *Evolutionary Adaptation:* An illustration of evolutionary adaptation is the peppered moth (*Biston betularia*). Originally these moths were light in color, which made them well camouflaged against lichen-covered, light-colored trunks of trees. In regions that became industrialized, industrial smoke darkened the tree trunks. Gradually the peppered moth population in industrial areas became predominantly composed of a dark variety, which was well camouflaged against the dark trees.
2. *Physiological Adaptation:* Many species can adapt to changes in environmental temperature: sweating, in man, is an example of adapting to heat changes.

3. *Sensory Adaptation:* If we are in a dark room and then the light is turned on, the eye adjusts to the change in a sensory stimulus, light intensity, by changing the diameter of the pupil.
4. *Adaptation by Learning:* This is a very general form of adaptation and is exploited in many ways. Animals can learn which food is most nutritious, where food can be found, which place gives the most shelter, and so forth.

Note that these different kinds of adaptations work on different timescales. Typically, sensory adaptation is the quickest, whereas evolutionary adaptation takes many generations. In this book, we focus mainly on adaptation by learning and through evolution.

Ecological Niches and Universality

DEFINITION

If we look at biological agents—animals—we find that they require a particular kind of environment for survival that is suited to satisfy their needs. Such an environment is called an animal's ''ecological niche''. Wilson (1975) defines ''ecological niche'' as follows: ''The range of each environmental variable such as temperature, humidity, and food items, within which a species can exist and reproduce'' (p. 317). It should be added to this definition that niche occupancy by a particular species usually implies competition. Different occupants of the niche compete for the same resources like food and space.

In nature, there is no such thing as a ''universal animal.'' Animals (and humans) are always ''designed'' by evolution for a particular niche. (We put the term ''designed'' between quotation marks to indicate that it is meant metaphorically: Evolution does not have a particular design goal.) Agents behave in the real world. As we pointed out, they always require certain conditions for their survival. A robot always requires some kind of energy source. It must be equipped with sensors and effectors in order to perform its task in a particular environment, or more precisely, in a particular ecological niche. To take the earlier example, if the robot has to work at night, it may be better to equip it with IR devices rather than with vision sensors. So, the idea of an ecological niche holds for robots as well (focus 4.1). It follows that there can be no universal robot, because the robot must perform in the real world, which consists of many varied environments to which a particular

Focus 4.1: A Market View of Robot Adaptation

David McFarland (1991), a leading ethologist and head of the animal robotics group at Oxford University, proposed an enjoyable analogy between ecological niche in animals and market niche in robots: ''Niche occupancy usually implies competition. When animals of different species use the same resources or have certain preferences or tolerance ranges in common, niche overlap occurs. This leads to competition between species, especially when resources are in short supply'' (p. 24). Just as animals occupy biological niches, robots occupy market niches: they are toys, cleaning robots, or whatever. A cleaning robot has to compete with human cleaners and other cleaning machines. The customer evaluates the performance of the robots and selects the ones that best fill his or her needs. This induces selective pressures which, in the end, determine whether a robot will ''survive'' in the marketplace. Table 4.1 provides an overview of the analogy between animals and robots (adapted from McFarland 1991, p. 24).

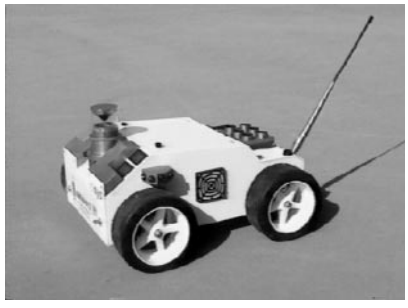
Table 4.1 Analogies between animal and robotic life cycles (from McFarland, 1991, p. 24).

	Biology (Animal)	**Market (Robotic)**
Return on investments	Number of offspring	Gross sales income assuming no failures
Reproductive probability	Chance of juvenile surviving to breed	Chance of product reaching the market
Development period	Age at breeding	Development cost
Design success (Rate of return)	Net rate of increase of genes (Fitness)	Net rate of increase of money invested in design (Instantaneous interest rate)

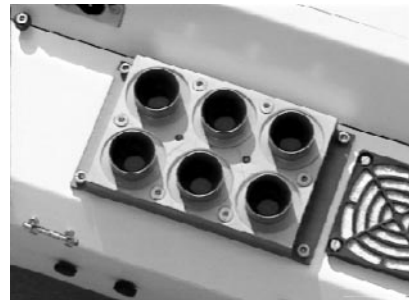
a

b

c



d



e

Figure 4.5 A robot designed for a particular ecological niche: (a) The desert ant *Cataglyphis*, (b) its niche, and (c) its navigation behavior—searching for food in a winding path, returning to the nest in a straight line; (d) the entire robot; and (e) the polarized light sensor module it uses for navigation. The Sahabot II (for *Saha*ra Ro*bot* II) has to operate in the Sahara Desert. Because its ecological niche is the desert, this robot is equipped with polarized light sensors and an onmidirectional camera (see figure 16.1). The robot is used for experiments to investigate the navigation behavior of *Cataglyphis*—more specifically, to evaluate different models of acquiring compass information from the polarized light pattern of the sky, and to test different models of visual landmark navigation. (Figures a, b, c by Rüdiger Wehner; reprinted with permission.)

robot may or may not be suited. Figure 4.5d shows an example of a robot, called the Sahabot (for *Saha*ra Ro*bot*) designed for a very special ecological niche, the Sahara desert (figure 4.5b). The Sahabot was developed to investigate the navigation behavior of the desert ant *Cataglyphis* (figures 4.5a and 4.5c).

This nonuniversality is quite in contrast to computation. As discussed earlier, computation is universal: Turing machines are the only machines that need to be studied. This is, of course, only possible because computation, by definition, takes place in a virtual world. And universality holds only in this virtual world. Computers are sometimes said to be universal, universal in the world of computation. If we look at computers as real machines, they depend very much on their environments. They need a continuous supply of electricity, they must be handled by their users with care, they must not be exposed to too much heat, and so forth. In that sense, computers, just like any other artifact, are designed for a particular ecological niche. Of course, some robots can exist in more different types of environments than others, so their niche is broader, but it is still there.

The fact that agents in the real world are not universal but have to function in a particular niche sounds like a severe restriction. But there is a lot of leverage to be gained by it, too. The fact that the ecological niche is restricted and has its own laws and characteristics, its types of objects, its types of agents, its temperature profile (i.e., how temperature changes over time), its lighting conditions, and so forth, can be exploited. Assume, for example, that in a particular niche only large objects are relevant. Then there is no need for a high-resolution sensor for distinguishing really small objects. If the niche is flat, wheels are sufficient. Often, learning problems that seem intractable at the purely computational level converge in real time if the constraints of the ecological niche are exploited. For example, if all objects of interest have a bilateral symmetry, as many living beings, this implies that learning can be restricted to one side, cutting computational costs in half. However, as always, there is a trade-off: The more constraints we exploit in our designs, the less universal the agent is. We return to this issue in chapter 13 when we discuss the principle of cheap design.

CHARACTERIZING NICHES

If we want to exploit the constraints of an ecological niche systematically, we also need a systematic characterization of niches, a

kind of taxonomy. Coming up with such a taxonomy, as it turns out, is not nearly the trivial matter it would first appear, because such chacterizations have to be made with respect to a particular agent, to its sensors and its motor system. Only those properties of environments matter that are behaviorally relevant. For example, to an ant, small pebbles, twigs, and puddles are behaviorally relevant —it can sense them and avoid them—whereas to an elephant, they are not—its sensory-motor system is not sufficiently fine-grained. Intuitively, one important distinction is whether the environment is static or contains objects that move on their own, such as other agents. Another concerns the size of objects, the distribution of food, circadian cycles, the roughness of the terrain, and so forth. Although such a taxonomy would clearly be important, it has so far resisted efforts to create it. Only a very few papers have even ventured into this topic area. One approach is to define environments by the constraints they satisfy. Horswill (1992) identified a number of ''habitat constraints.'' One example is what he defined as the ''background texture constraint.'' If the carpets or floors in a building have only fine-scale texture, from a distance, the floor appears uniform. If the illumination is uniform, then the areas of a camera image that correspond to the floor should have uniform brightness. Any deviation from this uniformity must therefore be an object. Horswill also defined the ''ground plane constraint.'' An environment satisfies the ground plane constraint if all objects in the environment, including the agent, rest on a single planar surface. Obviously, exploiting these constraints enormously simplifies vision processing. Office environments usually satisfy both of these constraints, as do some home environments, though some will have more textured grounds. We return to these constraints in chapter 10.

Another approach to classifying niches is to define environments by the predictability of the results of actions within the environment. Certain environments are more predictable than others; the less predictable an environment, the harder it is to design an agent for it. Thus, it would clearly be desirable, from the agent's point of view, to be able to characterize environments in terms of their predictability. (For more detail on this approach, see Wilson 1991.) The important factor in characterizing an environment is that it be done not in isolation, but with respect to an agent's complexity. We have more to say about this topic in chapter 13, where we discuss a particular measure of complexity.

In sum, for our purposes we use the terms ''complete agent'' and ''Fungus Eater'' to mean autonomous, self-sufficient, situated, embodied, agents designed for a particular ecological niche.

4.2 Biological and Artificial Agents

From our characterization of complete agents, it should be obvious that biological agents, animals and humans, fulfill all the criteria we set out: They are self-sufficient, autonomous, situated, embodied, and they are designed for a particular ecological niche. This is not surprising: The characterization was developed to explain natural intelligence. If creatures, including humans, had not met these criteria, they would not have survived in the first place.

Every psychologist, every biologist, in fact everyone in cognitive science, recognizes that in the best case, one would investigate complete agents and all their behaviors. However, from a methodological perspective it is not possible to study, for example, humans in all their intricacies. Thus, we must cut the problem down into manageable chunks. So even if we endorse a complete-agent view, we must make simplifications. The question, therefore, is not whether to make simplifications, but how to make them. In contrast to the classical way of modeling, in the embodied approach, the agents are ''cut up'' in a different way. An excellent illustration is the subsumption architecture that we discuss in chapter 7. The important point to be made here is that whatever aspect of intelligence we investigate, we must keep the entire agent in mind. This is not always easy to do, but it represents an essential design principle. It is summarized as design principle 1, the complete-agent principle, in chapter 10.

Our methodology for studying naturally intelligent systems is synthetic, meaning that we have to build artificial agents to mimic natural ones. The remainder of this chapter develops a basic framework for designing artificial agents.

Artificial Agents

In chapter 1 we mentioned three goals that we may want to pursue when building artificial agents:

1. building an agent for a particular task or a set of tasks
2. studying general principles of intelligence

3. modeling certain aspects of natural systems, that is, humans or animals

Goal (1) is from the engineering perspective, goals (2) and (3) pertain to cognitive science. All three goals are intimately related. In particular, goals (1) and (3) contribute to goal (2). We discuss these goals in more detail in chapters 16 and 17 when we discuss how to design and evaluate the agents we have built. For now we simply provide, as a very cursory review, a few examples illustrating goals (1) and (3), with the intention of providing an idea of what agent models can be used for.

The artificial agents we will design and study are of two types, robotic agents and simulated agents. Both are important tools. Some researchers have a preference for robots, others for simulation. We argue that both are needed, depending on the particular purpose of investigation.

ROBOTIC AGENTS

We now discuss a number of robots developed for various purposes. Let us first look at an example that illustrates the goal (1) above, the Mars Sojourner. Even though it was developed for a particular set of tasks (conducting experiments and collecting data on Mars), it nicely illustrates some of the fundamental issues such as autonomy, self-sufficiency (goal 2). We then turn to a few examples from biology to illustrate goal (3): cricket phonotaxis and human development and cognition.

Mars Sojourner

The Mars Sojourner has recently received a lot of attention in the media. Though today's robotic agents, in contrast to biological agents, do not fulfill all the criteria for complete agents that we set forth in section 4.1, the Sojourner comes relatively close. It is obviously embodied: It is a physical robot equipped with sensors and means of locomontion (wheels). It is self-sufficient, that is, it has to worry about its own energy supply: There is no human to exchange its batteries. It is also situated: The only means it has for acquiring information about its environment is its own sensory system. Further, it has a certain degree of autonomy, at least during real-time operation, though its autonomy is very limited, because most of its decisions are made by the mission control staff in the Jet Propulsion Lab in Pasadena. For instance, the ground staff decides

on what task the Sojourner is to execute next, what area it has to explore, what data it has to collect, and what pictures it should take. Focus 4.2 discusses the Mars Sojourner in more detail.

Cricket Phonotaxis

In chapter 1 we mentioned a robot built to model the phonotactic behavior of crickets (figure 1.10). Remember that by phonotaxis we mean those processes by which animals move toward a sound source, in this case the calling song of a potential mate. Our description here is short, just sufficient to make our point. (For details, see Webb 1993, 1994). Male crickets produce a particular sound by rubbing one wing against the other. Females can find a male by this cue over distances of 20 meters through rough vegetation. One would think that the cricket would need mechanisms for distinguishing the sound from the songs of other species and for analyzing the direction from which the sound is coming. It turns out that this is unnecessary because of the way phonotaxis works (Webb 1993). Instead of using a neural mechanism for recognizing the male's calling song, or an information process, the cricket uses a *physical* mechanism. Through this physical mechanism the irrelevant parts of all the sounds present in the environment are filtered out, so that only the ones concerning the calling song of the mate are registered by the cricket. Thus, without ''analyzing'' the sound, the cricket reacts only to the appropriate songs. This is an example of what biologists call ''matched filters.''

Webb's robot that models this phonotactic process in crickets has no legs and but two wheels. From this example it becomes clear that however close one tries to mimic a natural system, abstractions will always have to be made. This statement is generally true of models of any sort. Whether one considers Webb's model a valid one is a matter of the criteria to be applied and what one is interested in. Webb was particularly interested in the sensory-motor coupling and the theoretical question of the inseparability of perception from action. (We discuss how to evaluate models in chapter 17.)

Other examples of how robots are used to investigate biological agents are Franceschini's housefly navigation robot (Franceschini et al. 1992), and Lambrinos's ant navigation robot (Lambrinos et al. 1997; figure 4.5). Like Webb, these researchers have also made significant abstractions in constructing their robot models. For example, their robots are wheeled and much bigger than real insects.

Focus 4.2: Sojourner—The Mars Microrover

On December 4, 1996, NASA launched the Mars Pathfinder spacecraft from Kennedy Space Center. The spacecraft landed on Mars on July 4, 1997, and released Sojourner (figure 4.6), the first robotic roving vehicle to be sent to Mars. Sojourner is named after Sojourner Truth, an African-American reformist who lived during the Civil War; the name was chosen because it means ''traveler.'' Sojourner was built at the Jet Propulsion Laboratory of the California Institute of Technology in the southern California city of Pasadena. Sojourner's main function is to demonstrate that small mobile robots can actually operate on Mars. Sojourner is designed to conduct various science and technology experiments. For example, its cameras were used to take images from which a map of the landing site was constructed. Sojourner is unique not only because it is the first robot sent to Mars, but also because its total cost of development was only 25 million, a very low cost compared to that of previous interplanetary spacecraft, and also because its total development time was only three years.

Sojourner weighs 11 kg on earth and is 630 mm long and 480 mm wide. The ecological niche on Mars is a very rocky, uneven surface, and one major task of the NASA engineers was to equip the robot with means to operate in such a difficult environment: The robot therefore has six wheels instead of four: Six-wheeled robots can overcome obstacles three times larger that those that can be crossed by four-wheeled robots. Sojourner moves on its six

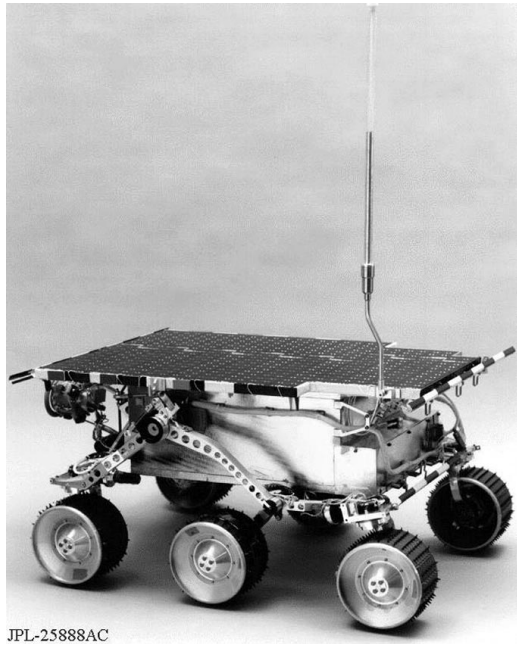


Figure 4.6 A picture of the Mars Sojourner (credit: NASA/JPL/CALTECH).

Focus 4.2 (continued)

wheels in a radius of about 10 meters around the spacecraft at speeds up to 0.6 meters per minute. Moreover, Sojourner's wheels and suspension system are built in such a way that the robot can tip up to 45 degrees as it climbs over rocks without falling over. Sojourner is equipped with a large number of sensors for detecting obstacles and hazards. Onboard sensors include simple bumper sensors for collision detection; cameras for imaging, distance calculations, and identification of target objects; accelerometers for hazard detection; and devices for measuring the speeds of the wheels (wheel encoders) that are used for estimating distance traveled.

Communication with the microrover, which is the general name for a robot of Sojourner's type, is accomplished via a radio communications system. The robot operates in a kind of supervised autonomous control. It receives remote commands from engineers on Earth instructing it where to go next. Commands are generated as follows: The camera system on the Pathfinder takes images of the robot. These images, together with additional images from the robot's cameras, are displayed on a computer at the control station on Earth. The engineers can designate goal locations on these displayed images. The robot then receives commands in the form "Go . . . ," which it executes autonomously while simultaneously avoiding obstacles and hazards. Communication with the robot does not occur in real time because it takes about 11 minutes for a signal to travel from Earth to Mars. This means that after the engineers have sent the instructions for the next goal location, the robot navigates there autonomously, that is, without human intervention. But it still has a very limited autonomy.

Like that of every other robot, Sojourner's equipment—computers, motors, communication system, sensors—requires power. The robot generates most of its power by means of a solar array that provides about 16 watts of power at noon on Mars, allowing the robot to perform most of its required tasks. In addition to this solar array, the robot is equipped with batteries that are needed when there is insufficient sunlight for the solar array to provide adequate power. Once depleted, these batteries cannot be recharged. Thus, redundancy has been built into the robot's power system: Should either the batteries or the solar array fail, the robot can still complete its tasks using the other power source. As discussed in Chapters 10 and 13, redundancy in design is very important. More detail on the Mars Sojourner can be found in Matijevic 1996 and Stone 1996.

Human Development and Cognition

Whereas some people would agree that robots can be used to model aspects of insect behavior, there is general skepticism that this can be done for human intelligence. However, a number of recent projects are highly promising. An ambitious approach is the Cog project at the MIT Artificial Intelligence Laboratory (e.g., Brooks and Stein 1993). The main goal of the Cog project is to study developmental processes from the very beginning by focusing on the sensory-motor aspects of intelligence using a complex humanoid robot. (Details of the project are given in chapter 7.) Experiments by Scheier and Pfeifer (Scheier and Pfeifer 1995; Pfeifer and Scheier 1997) demonstrate category-learning capabilities on robots interacting with the real world. Scheier and Pfeifer's working hypothesis is that ''high-level cognition'' can be achieved by having many, largely peripheral processes working simultaneously without central integrating mechanisms. This strategy is now pursued by a number of research labs around the world. (These experiments will be discussed in greater detail in chapter 12.) Yasuo Kuniyoshi, a leading robotics researcher at the Electrotechnical Laboratory in Tsukuba, Japan, near Tokyo, ventured to build a full-featured humanoid robot to conduct experiments on human development. The project is in its initial stages but holds great promise (e.g., Kuniyoshi and Nagakubo 1997). The point here is that just as it is possible to use robots to model insect behavior, we can use them to model human behavior. But the simplifications and abstractions are of a different nature (see chapters 16 and 17 for more detail).

Conclusions

None of the robots discussed in this section fulfills the criteria of a complete agent as discussed in section 4.1. The Mars Sojourner comes closest, but the Sojourner's autonomy is extremely limited: It is, in fact, deliberately kept within limits to minimize risk. Still, all the robots discussed in this section are, by the very fact that they are robots, embodied. They are also situated, in the sense that they interpret their environments from their own perspective. Some do have a certain level of autonomy: They are equipped with learning mechanisms that enable them to acquire their own history. They are not entirely preprogrammed. Their behavior depends on the situations they have encountered in the past. Finally, they are self-sufficient; only to a very limited extent. We believe that all the robot studies mentioned are highly valuable and provide impor-

tant insights, but we also see a need to investigate more complete agents.

SIMULATED AGENTS

It is, in principle, possible to simulate any physical process on a computer. As a consequence, it is possible to simulate any physical robot whatsoever: There are no restrictions. Let us look at some examples of such simulated agents.

Insect Walking

Randy Beer, a computer scientist with a strong interest in biology, developed a model of insect walking in simulation (Beer 1995) and used artificial evolution to study what sorts of gaits would evolve. He made many simplifications in his model. For example, the legs he employed were sticks without mass; that is, they had only one joint. Elasticity in the joints, friction, energy dissipation, and the like were ignored. In spite of these simplifications, Beer's simulated insect evolved to the point that it walked with very natural gaits that can be found in biological insects. Other agent simulation studies on insect walking have been conducted by prominent German biologist and neuroethologist Holk Cruse at the Center for Interdisciplinary Research in Bielefeld (e.g., Cruse et al. 1996).

Ant Navigation

Not only insect locomotion has been studied, but also insect navigation: how insects find their way to a food source and back. A famous example of simulation that took into account the situated character of the agent is the ''snapshot model'' by Cartwright and Collett (1983). The hypothesis to be tested in the models is that the insects, as they leave the nest, take some sort of image, a snapshot of their environment, to be used on their way back. The image is called a snapshot because it is thought to be relatively unprocessed. This idea is currently being vigorously debated.

Locomotion in Fish

Demetri Terzopoulos and his research group of the University of Toronto were interested in complex computer animations that would feature lifelike animals, such as, for example, fish. To achieve natural-looking movement, they decided to simulate not only the movements of the fish itself, but its physical interaction with the environment, the fluid dynamics as the fish is moving its body and its fins (Terzopoulos, Tu, and Grzesczuk 1994). Moreover, they modeled visual perception from an entirely situated

perspective. The movements achieved in this way look remarkably natural (see chapter 8). In the field of artificial life, agent simulations are very common.

Humanoid Interaction

The humanoid robot of Kuniyoshi mentioned earlier not only is being built as a physical robot, but is also being tested in simulation before the robot is constructed. This combined philosophy is used in many projects and is highly productive. Kuniyoshi and his colleagues have made a great effort to capture the dynamics (i.e., the physical forces) and not only the geometry (Kuniyoshi and Nagakubo 1997) of movement. Many simulations of robotic systems neglect dynamics or do not take them sufficiently into account.

Artificial Creatures

Simulated agents from the class of artificial life agents are used in studies of goal (2) discussed above, that is, to investigate principles of intelligence. Karl Sims has created a number of fascinating artificial organisms (Sims 1994a, 1994b). Not intended to mimic specific natural organisms. Sims' creatures ''live'' in a simulated physical environment: There is gravity, so the creatures have a certain weight, and there is friction. Moreover, similar to Terzopoulos's fish, fluid dynamics is modeled for creatures living in water. This environment is independent of the creatures themselves, which gives the simulation the strong flavor of real agent-environment interaction. This kind of simulation is becoming increasingly popular in virtual reality settings. We give a detail account of Sims' creatures in chapter 8, on evolution.

Real-World Robotic Agents and Simulated Agents

Our main interest in building autonomous agents is ultimately to improve our understanding of intelligence. There is an ongoing debate whether in order to achieve this goal, one can work with simulations or whether it is necessary to build real robots. To provide a short answer: Both are needed. The pros and cons are listed in table 4.2. At first sight, it seems best to use simulation because simulation is fast, cheap, and flexible. Closer inspection, however, reveals that a physically realistic simulation, which is often required, for example, when the results are to be tested on a real robot, is extremely hard to develop. Let us illustrate this point with two examples.

Table 4.2 Comparison of real robotic and simulated agents.

Criterion	Robotic agents	Simulated agents
PHYSICAL SYSTEM		
Agent	Must be physically built and run; great potential for breakdowns, slow, cannot be run in the absence of experimenter	Arbitrary number of copies can be produced; well-suited for systems involving many agents and artificial evolution; functions reliably even in the absence of the experimenter
Physical environment	Given; environment has its own dynamics	Everything must be taken into account by programmer; often hard to simulate; realistic simulations computationally expensive
Sensors	Given; no idealizations, no ''cheating''; often unanticipated effects occur (interference, reflectory properties of surfaces, drastic changes in intensity)	Sensors hard to simulate realistically; idealized sensors common, e.g., distance, object or agent recognition
Motor system	Dynamics given; complex ones hard to build and hard to control; imprecisions	Dynamics hard to simulate realistically
Dynamics in general	Given; exploitation of dynamics necessary and natural (cf. the passive dynamic walker, chapter 13)	Hard to simulate; often ignored in simulations; dynamics often not exploited
RESEARCH		
Emergent phenomena	Indefinite richness of physical environment offers great potential for emergence	Emergent phenomena frequent, but limited to basic specification present in simulation
Effort required	Can be considerable; experiments take a long time; experimenter must be present; debugging is hard	Effort to develop physically realistic simulations considerable; experiments can be run easily; presence of experimenter not required; changes quickly realizable
Gaining insights (heuristic value)	Highly productive	Highly productive

Table 4.2 (continued)

Criterion	Robotic agents	Simulated agents
Abstractions	Significant and obvious	Significant but less obvious
Scaling to more complex systems	Sensory systems are relatively easily made more complex; motor systems are much harder	Highly complex robotic systems are often not simulated; rather, abstractions are introduced (e.g., a grasp operation as a given elementary action)
Artificial evolution	Only possible for control architecture, not for complete robots	Simulation currently the only possibility; many surprising effects
Agent societies	Currently significant effort to build multiple robots (restricted to small numbers); all sensor processing based on real sensory inputs	Easy to simulate; duplication of agents trivial; idealized sensors (e.g., for object recognition) easily introduced

First, IR sensors are often used to measure proximity (nearness) to an object. But in fact, IR sensors yield an accurate measure of proximity only under unrealistic conditions: IR sensor are active sensors, that is, they send out an IR signal and measure the intensity of the reflected IR light. This creates several problems. First, the amount of light reflected depends on the properties of the materials in the environment. Second, a particular IR sensor cannot distinguish between its own IR signal and those coming from other sensors. And third, sunlight and artificial light contain IR light, which the sensor also measures.

Second, physical robots have mass, and gravity acts on them automatically as it does on any object in the real world (figure 4.7). If we want our simulated robot to have mass and weight (i.e., gravity acting on it), we must explicitly introduce it into the simulator. If a robot has the task of moving around in an office space without getting stuck, one strategy for accomplishing this is to exploit its own inertia to get out of impasses. By rushing into objects with relatively high speed, the robot bounces off, slides around and, very often by chance, faces in a direction in which it can move forward again. This process, which in the real world simply happens, would be extremely hard to capture formally in a simulation.

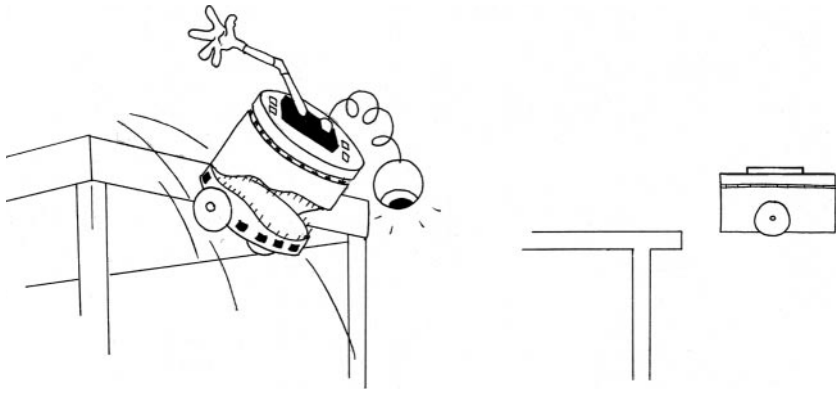


Figure 4.7 Comparison of real world and simulation. In both cases, gravity has not been programmed into the system. In the real world (a), the robot drops to the ground anyway—gravity is part of the real world and does not have to be programmed. In the virtual world (b), the robot moves off the edge of the table and does not fall, making the simulation a poor representation of real-world events in this case.

Abstractions

Let us stop and summarize what we have said so far in this section. Whenever we are making a model, robot, or simulation, we have to make abstractions. As pointed out above, the insect robots, that is, the cricket and the ant robots, have wheels instead of legs, have electrical motors instead of a carbon-based physiology, and are much bigger and heavier than real insects; the ant robot only has three polarization elements (rather than about 200, as the real ant). Still, the claim is that the robot models reproduce interesting aspects of insect navigation. In building a model, we have to choose a level of abstraction, a level at which we are comparing the biological system and the robot model. Note that the robot model is not only a model, but a behaving system itself that can be studied in its own right. Beer's walking insect, for example, has six massless sticks as legs—a potential source of error.

Implicitly, we are assuming, when we build robot models of insects, that the navigation mechanisms of the insects are not influenced by the means of locomotion, the size, and the body weight, to mention just a few of the assumptions we make. We have to be aware of the fact that these may turn out to be blatantly false. On the other hand, we have fully embodied and situated systems: all the information about the environment is acquired through the models' sensory systems in the interaction with their environments. The models do have a certain level of autonomy: The Sahabot can acquire some information about the environment, and

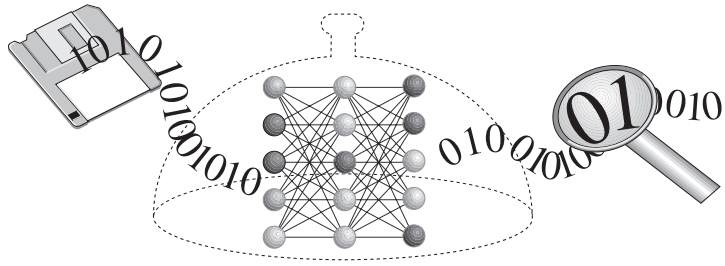


Figure 4.8 The principle of operation of the ALCOVE model. The model receives its input data from a file prepared by an experimenter (illustrated by the diskette). This input is used for learning. The model has no real interaction with the environment. A human (illustrated by the magnifying glass) must interpret the meanings, of the bit strings produced by the network.

its later behavior depends on this information. However, this autonomy is limited. The last property, self-sufficiency, is not characteristic of any of the models. Thus, we are excluding an important consideration from our models.

Another assumption in creating these insect models is that the insects' navigational mechanisms are independent of energy supply. This, once more, may turn out to be false. Although we consider this to be unlikely, we have to keep it in mind and be prepared for it.

Agent Simulation versus Classical Simulation

So far we have been talking about agent simulation, which is concerned with the simulation of a complete agent with as many of its essential characteristics as possible (embodiment, self-sufficiency, situatedness, autonomy). This contrasts with the more classical style of simulation, in which certain aspects of an agent's behavior are simulated in isolation. The differences are best illustrated with an example.

In psychology, connectionist models have become very popular. A prominent example is the ALCOVE model of categorization (Kruschke 1992), explained in more detail in chapter 12. Here we focus on the differences between this model and agent-based models. The point is not to critize this particular model—which in fact explains the results of many psychological experiments—but rather to point out the limitations, from an embodied cognitive science perspective, of connectionist models in general. The schematic overview in figure 4.8 shows the essential differences between connectionist and agent models.

In the ALCOVE model, there is an input, an intermediate, and an output layer (the category layer). The data are provided by the model designer: the model reads one input vector after another and processes it. In contrast to agent simulations, the model has no direct interaction with its environment. One important implication is that the model's output has to be interpreted by the designer and does not lead automatically to the next input. In agent-based models, the loop from input to output to input is closed; so there is no human intermediary in the loop. This characteristic is highly constraining—errors in the output lead to subsequent erroneous input patterns; the model has to be consistent with respect to its own outputs—and can be exploited in various ways. (In fact, we devote chapter 12 to mechanisms that allow an agent to structure its own input by interacting appropriately with the world.) Finally, the ALCOVE model processes all data it receives; it does not have to determine which of the data are relevant. In agent models, one of the hard tasks is to determine which of the continously changing input data should be considered relevant by the agent, for example, for learning. This book focuses on agent simulation and, of course, real-world physical agents.

We have looked at the kinds of agents that we want to build. Let us now look at how to go about designing agents and how to conduct experiments using the synthetic methodology.

4.3 Designing for Emergence—Logic-Based and Embodied Systems

This entire book is about design. In this chapter, we lay out some of the groundwork for design. The considerations outlined in this section are fundamental to every design effort, and getting them right from the beginning can help you avoid a lot of confusion and fundamental problems later on. The kinds of considerations relevant for agent design and design of classical systems are very different, as we will see shortly. In this section we use examples from two areas, medical diagnosis and agent design, to illustrate both so-called domain ontologies and low-level specifications. We also use the term ''high-level ontologies'' to clearly distinguish these from low-level designer commitments.

The section proceeds as follows. We first discuss classical design, starting with high-level concepts. We then introduce agents and show that the commitments involved in designing agents must be made at a different, lower level: What we are really interested in is adaptivity, which requires diversity and emergence. The art of

agent design is design for emergence, as Luc Steels (1991) has called it: Make design commitments that leave room for emergence of behaviors as the agent interacts with its environment. Throughout the book, we refer to emergence, a concept that we have already introduced and briefly discuss again below.

The Frame-of-Reference Problem in Autonomous Agent Design

Whenever we are involved in designing an intelligent system, we have to be aware of the frame-of-reference problem. As we discussed in chapter 3, the frame-of-reference problem concerns the relation between the observer, the designer (or the modeler), the artifact, the environment, and the observed agent. The artifacts that we study in embodied cognitive science are autonomous agents, but the argument holds for computer programs as well. Again we emphasize, because we can hardly overstress it, the importance of getting this problem straight from the very start. Our outline of the problem is based on Clancey's (1991a) extensive treatment. The frame-of-reference problem has three main aspects:

1. *Perspective issue:* We have to distinguish between the perspective of an observer looking at an agent and the perspective of the agent itself. In particular, descriptions of behavior from an observer's perspective must not be taken as the internal mechanisms underlying the described behavior.
2. *Behavior-versus-mechanism issue:* The behavior of an agent is always the result of a system-environment interaction. It cannot be explained on the basis of internal mechanisms only.
3. *Complexity issue:* The complexity we observe in a particular behavior does not always indicate accurately the complexity of the underlying mechanisms.

Let us briefly illustrate these points with a famous example, Simon's ant on the beach.

SIMON'S ANT ON THE BEACH

Simon (1969) has used the metaphor of an ant to illustrate some basic principles of behavior; here we use his metaphor to illustrate the three aspects of the frame-of-reference problem. Let us assume that an ant starts on the right and its nest is somewhere on the left. So it travels roughly from right to left. Figure 4.9 shows a typical path the ant might take. From the perspective of the observer, the path is seen as a trajectory on the beach between pebbles, rocks,

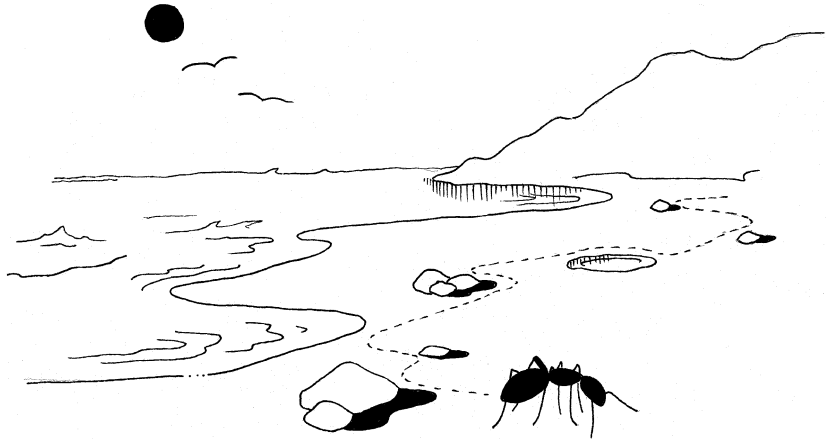


Figure 4.9 Simon's ant on the beach. Herbert A. Simon suggested that an ant walking on the beach illustrates that behavior that looks complex to an outside observer may in fact come about by very simple mechanisms.

puddles, and other obstacles. From the perspective of the ant, the world looks completely different because of its entirely different embodiment (different sensors, different brain, different body): To the ant, there are no pebbles, rocks, and puddles as we see them. This illustrates the perspective issue.

What the observer sees as a complex path is the result of the ant's behavior, that is, of the interaction of the ant with its environment. How does this behavior come about? It would be a mistake to assume that the entire path of the ant is stored in the ant's brain and then used to guide its behavior. More likely, the mechanisms driving the ant's behavior are actually very simple, implementing ''rules'' that we could describe as follows: ''if obstacle sensor on left is activated, turn right (and vice versa).'' (These rules are, of course, implemented in the ant's neural structures). This illustrates the behavior-versus-mechanism issue: behavior must be clearly distinguish from internal mechanism.

The behavior-versus-mechanism issue is directly related to the complexity issue: The trajectory, the result of the ant's behavior, looks complex to an outside observer, but in fact it came about by applying simple rules.

The point is that the complexity of the ant's trajectory emerges from the interaction of the ant with its environment, not from the internal mechanisms alone. Therefore, the complexity of the environment is a prerequisite for the complexity of the ant's behavior. To further illustrate this point, let us assume that we increase the size of the ant, say, by a factor of 100, and let it start in the same

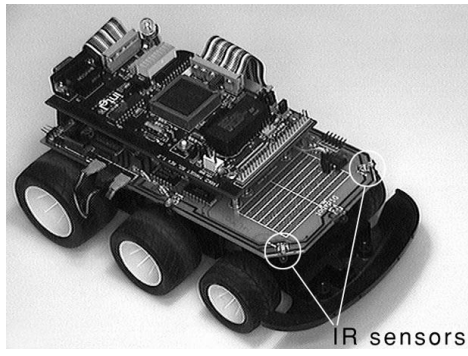
location with exactly the same behavioral rules as before, it would go more or less in a straight line! What appeared to the normal ant as obstacles would no longer be obstacles for the giant ant, whose sensors would not be sufficiently fine grained even to detect the irregularities of the beach. Thus in order to fully explain the ant's behavior, we need to take the internal mechanisms, the environment *and* their interaction into account. Behavior cannot be reduced to internal mechanisms, i.e. it cannot be explained on the basis of internal rules alone. We must take the agent's body into account; Changing the body leads to different behavior.

An example from robotics that also demonstrates the dependence of the behavior on the embodiment concerns the position of the sensors. Figure 4.10a shows a Didabot, a very simple kind of robot used for classwork exercises. In this experiment, only two IR sensors are used. The position of the sensors is shown in figure 4.10b. The control architecture consists of a very simple neural network that implements the rules of Simon's ant on the beach: If sensory stimulation on left, turn right; if sensory stimulation on right, turn left. This leads to obstacle avoidance behavior. However, if the robot encounters an object head-on, it pushes it, because it gets no stimulation from its sensors. If we now change the position of one of the sensors by moving it to the front (figure 4.10c), the pushing behavior disappears, (the robot will either turn left or right) even though exactly the same neural network was used. This illustrates the general point that the neural substrate of any agent can be understood only in the context of its embodiment.

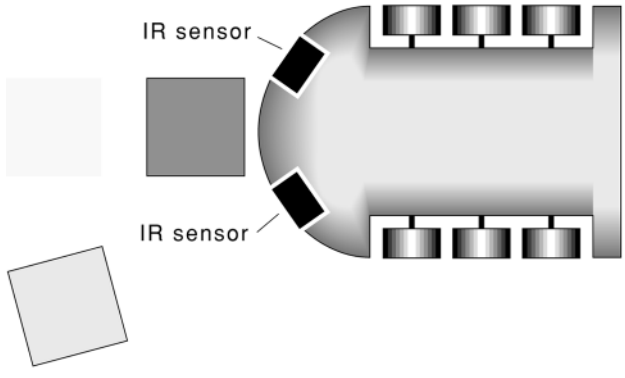
BUILDING A MODEL OF THE ANT'S BEHAVIOR

Let us further illustrate the frame-of-reference problem by looking at how a biologist might go about understanding the behavior of Simon's ant on the beach. Assume the biologist employs a synthetic approach; that is, he tries to understand the ant's behavior by building a model capable of reproducing certain aspects of its behavior.

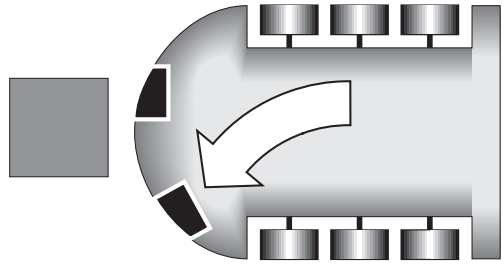
The most straightforward approach he could take would be to suppose that the trajectory of the ant is stored in its head, represented, for example, as some kind of network structure (figure 4.11a). This trajectory can be used as a plan for generating behavior: To find its nest, the ant simply replays the trajectory. Note that the biologist is making a category error: He is confounding a description of behavior (the trajectory) with the internal mechanism. To test the model, he now wants to use it to control a robot.



a



b



c

Figure 4.10 Illustration of embodiment. (a) The Didabot. (b) Sensor configuration 1. (c) Sensor configuration 2. Sensor configuration 1 leads to pushing and obstacle avoidance behavior, whereas sensor configuration 2 leads to obstacle avoidance only. Both configurations use the same internal neural control mechanism.

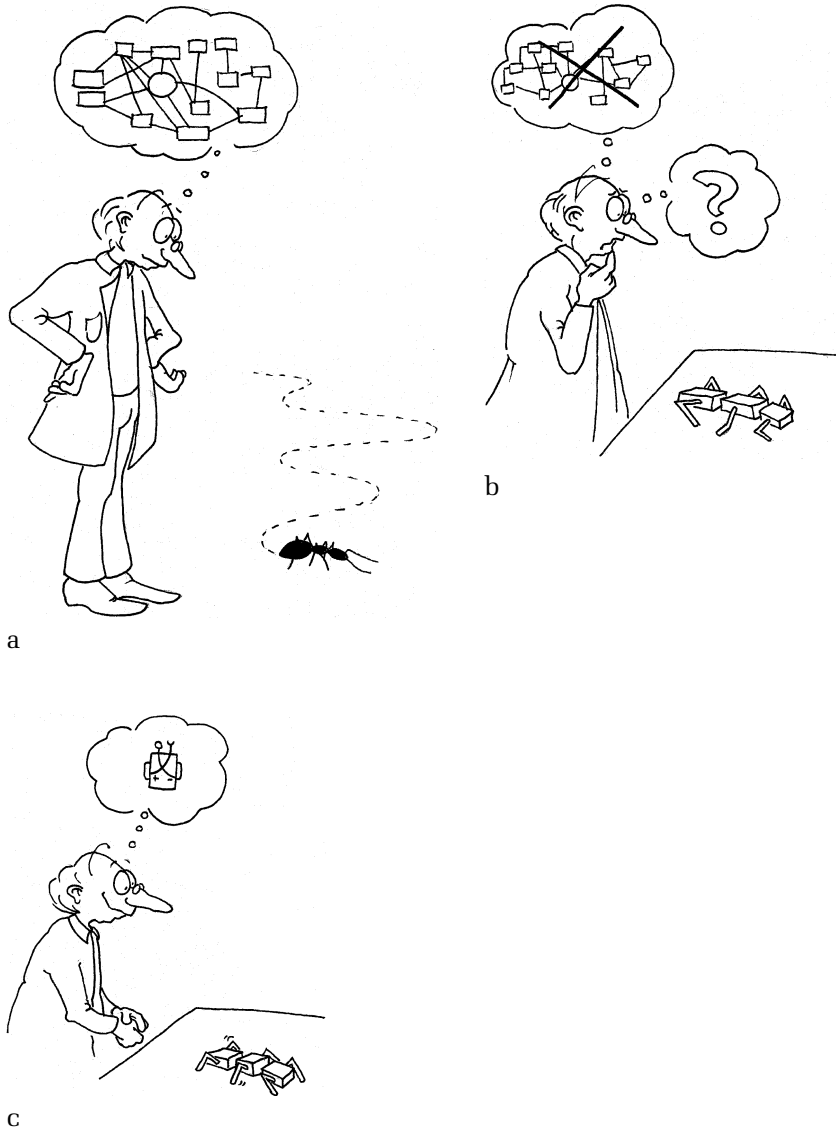


Figure 4.11 A biologist trying to understand the behavior of an ant. (a) First, he develops a model that directly maps the behavior onto an internal model. This illustrates the perspective issue. (b) Then he tries to use this model to control a walking robot. He discovers that it does not work well—the robot does not move. In other words, the model he hypothesized in (a) does not lead to the desired behavior. This illustrates the behavior-versus-mechanism issue. (c) Next, he realizes that a much simpler network will lead to the desired behavior. This illustrates the complexity issue.

This does not work very well (figure 4.11b) because of the category error. Because behavior is the result of a system-environment interaction, it is of little use to record past behavior and employ it to generate future behavior. If there is even the slightest of changes in the environment, the plan no longer works. This illustrates the behavior versus mechanism issue and the perspective issue. Behavior is something different from internal mechanism; it can be observed by an outside observer, whereas the mechanism is internal to the agent. Because of these considerations, the biologist realizes that a different kind of mechanism is required, and to his delight he finds that it is much simpler than the previous one (figure 4.11c).

We have deliberately chosen to illustrate the frame-of-reference problem with two somewhat whimsical examples, the ant on the beach, and the hypothetical biologist building a model of the ant's behavior. Here we only wanted to provide an intuition of the issues involved; the application of the problem to the scientific study of intelligence follows later.

High-Level Domain Ontologies and Low-Level Specifications

The title of this section may sound a bit cryptic but the basic idea is actually very simple. Whenever we design a system, we have to define the basic concepts or components, the *primitives*, that the system will use. For classical systems, databases, or AI systems, a *high-level ontology* or domain ontology has to be designed. It contains items such as, for a database system, a personnel record (with fields for name, age, sex, salary, department, projects, address, etc.), or for a medical system, symptoms and diseases. When designing an agent that has to interact with the real world, however, this no longer works. Designer commitments can no longer be made at this level—otherwise the designer runs into all sorts of problems, such as the symbol-grounding problem, to mention only one particularly thorny one. For an agent in the real world, design commitments have to be at a lower level, concerned with the agent's physical setup, its body, sensory, and motor systems. Whatever the agent learns about its environment should then result from the agent's interaction with the environment. We call these designer commitments a *low-level specification*.[2]

[2] We prefer the term ''low-level specification'' to ''low-level ontology'' because ontology triggers associations with logic-based systems.

HIGH-LEVEL ONTOLOGIES

Let us now be a bit more precise with some definitions. We use the term *ontology* very simply, in the standard way of the artificial intelligence literature (e.g., Russell and Norvig 1995, p. 222). A *domain* (or *high-level*) *ontology* has three essential characteristics:

1. It designates the basic vocabulary, the primitives, that are going to be used in designing the system. These are the only components that can be used: Everything in the system is built on top of these basic elements.
2. The meaning of these primitives is assumed to be given and shared by those involved, that is, the designers and the users.
3. The domain ontology remains constant for an extended period of time, often for the entire life of the system.

Thus, a domain ontology is a systematic account—a list—of all the basic concepts (i.e., the objects, relations, and operations) that are needed in a particular domain. The primitives have to be defined for any system whatsoever, be it a database system, a communication system, an expert system, a system for understanding natural language, or a robot. However, the kinds of primitives employed for computational systems and robots differ considerably. In a medical expert system—a computational system—they might include symptoms (red spots on skin, fever, diarrhea), patient characteristics (age, race, history), diagnoses (organisms, diseases), medical procedures to be applied (tests, treatments, therapeutic programs), and medical knowledge combining the concepts (bacterial meningitis is a subclass of meningitis). For each of the attributes within the primitives, all possible values have to be given. For example, for the attribute ''red spots,'' the values could be ''absent,'' ''present,'' ''strongly present.'' Table 4.3 offers a highly simplified sample domain ontology for a medical system.

All that the system to be designed will be able to do springs from and depends on this set of primitives initially specified by the designer. A state is a description of the current situation in terms of the primitives of the domain ontology. By means of the rules of inference, states are transformed into other states. For example, the state described by high fever, muscle pain, and high sensitivity to light, might be transformed into a new state called ''flu.'' In this perspective, learning—that is, the formation of new concepts—consists only of combining basic components or compound concepts in different ways. As an example, recall the robot JL that we designed in chapter 2. It combined the basic concepts ''green,''

Table 4.3 A simplified high-level domain ontology for a medical expert system. (To keep the example simple, the ontology here is based entirely on intuition and should not be taken seriously from a medical point of view.) Realistic medical systems can contain hundreds and even thousands of components in their domain ontologies.

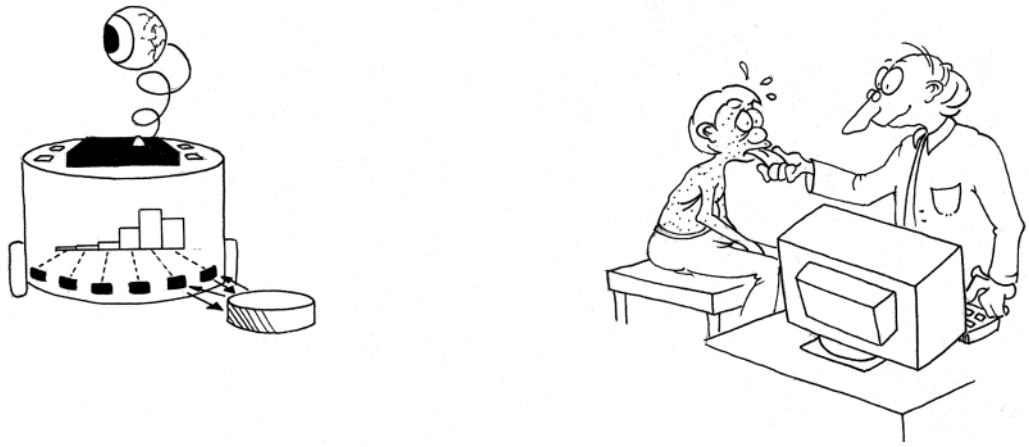
Category	Attributes
Symptoms	• Red spots on skin (absent, weakly present, strongly present) • Fever (none, weak, strong; alternatively: °C) • Diarrhea (absent, present, strongly present)
Characteristics of patient	• Age (a number) • Race (Caucasian, Indo-European, Pan-Asian, Semitic, etc.) • Weight (a number) • History (medical history)
Diagnoses	• Organisms (bacteria, viruses) • Diseases (influenza, pneumonia)
Medical procedures	• Tests (blood tests, growing cultures, urine tests) • Treatments and therapeutic programs (cures, diets, operations, physical therapy, psychotherapy, medication, radiation, etc.)
Relations, medical knowledge, problem solving methods	• Bacterial meningitis is a subclass of meningitis • Heuristic classification • Hypothesize and test

''ripe,'' and ''apple'' to form the compound concept ''Granny Smith.'' Here is another example: If we want to develop a natural language processing system that understands stories about restaurants (e.g., Schank and Abelson 1977), we must have an ontology that includes, for example, the components used in the restaurant script shown in figure 2.7, either as part of the ontology itself, or as concepts accessible by combining more basic parts. An ontology for a restaurant would have to contain elements like glasses, cups, tea, coffee, beer, serving, checks, eating, and so forth, again either as elements, or as compound concepts made up of more basic components.

Ontologies at the computational level are well defined because they have their origin in logic. The situation is much messier in the case of robots, in which we have to define low-level specifications.

LOW-LEVEL SPECIFICATIONS

Above we defined a domain ontology as the vocabulary, the primitives that will be used in the design of the actual system. For clas-



a b

Figure 4.12 Comparison of high-level ontologies and low-level specifications. For the robot (a), there is no ambuiguity about the amount of stimulation at the sensory level, whereas the doctor (b) has a lot of room for interpreting whether red spots are present in the patient.

sical systems, it is fairly easy to decide at what level to designate the domain ontology. It is much less clear, however, at what level these primitives should be designated in the case of a robot. Obviously the robot's body, its sensory system, and its motor system have to be designed. Moreover, the individual components have to be connected in appropriate ways. Table 4.4 provides an overview of the components for a low-level specification of robots. The table's second column provides an abstract characterization in terms of states; the third suggests possible implementations.

As an example of a component in a low-level specification, let us take a standard vision sensor which is normally realized as a camera. What are its basic characteristics? It contains a number of light-sensitive cells. These cells can be in various states that are determined by physical processes, that is, the intensity of light registered at the cell. The output of the cell, that is, the signal produced by it (to be further processed), is roughly proportional to the light intensity. In other words, the interpretation of the signals from the light-sensitive cells is straightforward.

By contrast, attributes of high-level ontologies are often open to a great deal of interpretation. For example, what does ''red spots (weakly present)'' really mean? When do we talk about red spots? How red do they have to be? How big do they have to be? How dense is ''weakly present''? As a consequence of the great room

Table 4.4 A simplified low-level specification for a robot. The second column provides an abstract characterization in terms of states; the third suggests possible implementations.

System	Component	Characterization	Typical implementations
Body	body (without sensor and motor system components)	shape, weight, size, rigidity points of attachment for sensory and motor components	rigid frames (wheeled robots) multisegment flexible (humanoid robots)
Sensory system	visual sensors	light-sensitive cells (states: on-off, grayscale, color)	camera
	proximity/distance	sensor readings related to distance (states: number of different readings)	IR, ultrasound, or laser range-finder sensor
	touch	requires physical contact (states: on-off)	microswitch; saturation of IR sensor; skin sensors
	speed sensors	sensor stimulation related to speed (states: number of different readings)	wheel encoders (wheel turns); optical flow
Motor system	wheel drive system	speed and direction of wheels (states: speeds, steering angles)	wheels driven individually by electrical motors
	leg locomotion system	(states: joint angles, forces)	forces supplied by electrical motors
	arm	(states: joint angles, forces)	forces supplied by electrical motors
	body motion system	(states: joint angles, forces)	forces supplied by electrical motors
Interactions among components	mechanical	type of connection between mechanical parts	mechanical connections always (implicitly) given
	electrical	types of signals that can be exchanged within the robot	bus system connected via a microprocessor; separate physical connnections possible
	electromagnetic	components interact without a wire connection	given by physical system; not deliberately designed
	thermal	interactions through materials surrounding a component	given by physical system; not deliberately designed
	environment	not explicit	given by system-environment interaction

for interpretation, such systems always require a human for their operation; in fact, they require a human expert, as figure 4.12 illustrates.

Many more sensors could be added to table 4.4 (torque sensors in the joints, position sensors, flow sensors, temperature sensors, etc.). The particular choice of sensors depends on what the designer intends to use. The position of the sensors on the robot is also an essential part of the low-level specification.

Let us now look at the motor system for a moment. Just as on the sensory side, the ways in which motor systems can be designed are virtually unlimited. Take a legged robot. Its legs have joints that can assume different angles, and various forces can be applied to them. Depending on the angles and the forces, the robot will be in different positions and behave in different ways. Further, the legs have connections to one another and to other elements. The details of how the various elements are connected are not important for here, but it is important to note that these connections are often not made explicit in the specification, though they are essential for the robot's performance. If a six-legged robot lifts one of its legs, this changes the forces on all the other legs instantaneously, even though no explicit connection needs to be specified. The connections are implicit: They are enforced through the environment, because of the robot's weight, the stiffness of its body, and the surface on which in stands. Although these connections are elementary—and the robot's behavior builds on them—they are not explicit in the low-level specifications, although they could be made explicit and included if the designer wished. Connections may exist between elementary components that we don't even realize. Electronic components may interact via electromagnetic fields that the designer is not aware of. What is normally explicitly designed are wire or data bus connections. So we see once again that because robots, bodies, sensor systems, and motor systems are real physical entities, it is not possible to define neatly what belongs into a low-level specification, certainly not as neatly as we can define the components of a high-level ontology. Moreover, the agent has a body with a particular shape, and it is not clear how shapes should be generally described.

We mentioned that the communication between the legs of a robot can be implicit. As a general rule, much more is implicit in a low-level specification than in a high-level ontology, simply because the physical world is always a given and it has its own properties,

irrespective of whether a designer is fully aware of them. Here we are encountering a fundamental implication of simulated agents versus real agents: In simulated agents, only what is made explicit exists, whereas in the real world, many forces exist and properties obtain, even if the designer does not explicitly represent them.

The Sensory Space, the Motor Space, and the Sensory-Motor Space

The notion of *sensory space* denotes all possible configurations of the sensory states. If we have a black-and-white camera with only two intensity levels (activation or no activation) and a 100×100 image, that yields a sensory space with 2^{10000} possible states. (There are 10,000 sensors, each having two possible states.) Remember that 2^{10} is roughly 1,000, so we have approximately 10^{30} different states. If instead of just these two intensity levels, we have 256 different gray levels, this yields an incredibly large number of states. We do not discuss the implications of this here, but simply point out that this very large number of possible states is a prerequisite for the generation of diversity (in other words, for adaptivity). Similarly to the sensory space, the *motor space* can be defined as the ensemble of possible states the motor system can assume, given a particular low-level specification. In this book we rarely look at sensory and motor systems in isolation: we normally consider the entire sensory-motor space, which denotes the entire range of possible configurations of sensory and motor states together. Logic-based systems, such as expert systems or natural language processing systems for written text (in electronic form), have no sensory space in the same sense robotic systems do, simply because they lack sensors. Nevertheless, we can define the sensory space for logic-based systems as the set of potentially different inputs the system can accept. This is precisely given by the domain ontology. Anything not predefined in the ontology (or not combinable from the elements of the ontology) cannot be presented as input to the system. Defined in this way, the sensory space (or better, the *input* space) is typically much smaller for an expert system: there are only the predefined concepts, and the values they can assume are restricted (e.g., the concept ''red spots'' can have the values ''absent,'' ''weak,'' ''clearly present,'' or ''strongly present''). Moreover, the number of basic concepts in such an input space is comparatively small, on the order of a few hundred. The input space can still be of considerable size, leading sometimes to combinatorial problems, but it is normally considerably less than 2^{10000} (and

that's a very simple case). Complexity in expert systems (and logic-based systems in general) is therefore computationally manageable.

From this discussion it follows that a system always communicates with its environment—including other agents—through its primitives. If we want to put a request to a database system, we can do this only by using terms that are already defined in the system, that is, terms either contained in the basic domain ontology or combinations of the latter. The same holds for the output of the system. If we want to interact with a robot, it has to be via components of the low-level specification.

Emergence

Our goal is to design agents that display emergent behaviors. The term *emergent* is used mainly in three different ways. First, it is often applied to situations, agent behaviors, that are surprising and not fully understood. Second, it refers to a property of a system that is not contained in any one of its parts. This is the typical usage in the field of artificial life, dynamical systems, and neural networks for phenomena of self-organization. Third, it concerns behavior resulting from the agent-environment interaction whenever the behavior is not preprogrammed. It is thus not common to use the term if the behavior is entirely prespecified like a trajectory of a hand that has been precalculated by a planner. Agents designed using high-level ontologies have no room for emergence, for novel behaviors. High-level ontologies are therefore used whenever we know precisely in what environments the systems will be used, as for traditional computational systems (like an accounts payable–accounts receivable program) as well as for factory robot systems. In unknown environments, a better strategy is to define the low-level ontology, introduce redundancy—and there is a lot in the sensory systems, for example—and leave room for self-organization. The following question immediately arises: Given a set of desired behaviors, how do we design the agent so that these behaviors will be emergent? How does design for emergence work? Chapter 16 discusses these topics; in chapters 11 through 13, we show concrete examples of how we can actually design for emergence.

Novel Situations and Novel Actions

In chapter 1, we saw that one of the important aspects of intelligent systems is adaptivity, that is, the ability to perform in novel situ-

ations. This implies on the one hand recognizing that a situation or environment is novel, and on the other generating new behavior appropriate to the now-changed situation. Let's investigate this point a little further.

''Computers can act only in situations that have been predefined by humans!'' computer skeptics often assert, ''and this is why computers cannot be used in environments in which there may be potentially novel situations.'' Computer enthusiasts reply: ''No problem. If a situation is encountered that has not been predefined, the computer simply displays a message on the screen saying something like 'no information available,' in which case the human operator can handle the situation.'' We can use the idea of domain ontologies to define more precisely what is meant by ''predefined'' and ''novel.''

Take our medical expert system. If the system encounters a patient with a combination of symptoms, say red spots, fever, liver pain, and a broken leg, and there is no rule that covers that particular symptom pattern, the system might display the message ''no information available,'' and the physician could take over. Such a case presents no problem. All the symptoms involved have been predefined; certain combinations have not been foreseen, but such cases are covered by the domain ontology: In these cases, a pertinent message can be displayed. However, if a symptom is not predefined, the system does not even recognize that it is faced with something new, and that does present a problem. Another example is a system for some type of process control: If there is no temperature sensor, the system—quite obviously—cannot sense temperature. So if the temperature rises above an unacceptable level (a novel situation), the system does not even know that it is a new situation because it does not ''know'' anything about temperature. Note that precisely the same point holds for robots, and for animals and humans, for that matter. Anything they can learn is constrained by the basic primitives, the low-level specifications. The reason humans can recognize truly novel situations is because of the large redundancy contained in their sensory systems. This point is of fundamental importance, and is incorporated as a design principle, the redundancy principle (see chapters 10 and 13).

A Hybrid Specification

We have discussed high-level ontologies and low-level specifications. We have also said that agents should first be designed by

Figure 4.13 A Japanese robot serving tea (from Kurzweil 1990, p. 319). The robot has to know about tea, teacups, saucers, the properties of liquids, and serving. But it also has to recognize and manipulate them through its sensory-motor system, its hardware. (Picture by Georg Fischer; reprinted with permission.)

defining the low-level specifications and then use mechanisms of self-organization. But why not have both a low-level specification and a high-level ontology on top?

Assume that you have the task of developing a robot to serve tea in a restaurant, like the Japanese robot in figure 4.13. Because you have to design a robot, you need a low-level specification that lists your commitments about the robot's physical setup and the potential connections between the components. Moreover, the robot needs to know about tea, teacups, saucers, properties of liquids, and serving, so you may want to include those concepts in its domain ontology. If you do this, you are defining a high-level ontology that implies a designer-based categorization of the real world. So there are now two levels at which you, as a designer, are making commitments. This introduces a new problem: the two levels have to be compatible. Achieving this compatibility has turned out to be extremely difficult, as the problems with model-based computer vision show (e.g., Tistarelli 1995). Moreover, defining a high-level ontology on top of a low-level one entails the symbol-grounding problem that we discussed in chapter 3. Thus, if the agent is to be situated and adaptive, it must learn about the environment as it is interacting with it, thus it is nonstatic. This

nonstatic bottom-up component must then match the high-level concepts. This is a notoriously hard problem to solve, because if implies solving the symbol grounding problem. But what should we do then, if we want to design a tea-serving robot? This is a fundamental research issue, and the interested reader is referred to issue 4.1 at the end of the chapter.

To conclude, the idea of this section has not been so much to map out a general low-level specification for robot design. From what we have said so far, it should be clear that it is not possible to define low-level specifications as clearly as high-level ontologies. Instead, the section has stressed the distinction between high-level and low-level design decisions. Low-level specifications make no mention of high-level categories corresponding to what we, as observers, would call objects (coffee cup, saucer, tea, beer, etc.). As we argue later, if concepts are going to be grounded, they have to emerge from this low-level specification, and the way of proceeding that we suggest does not work with high-level ontologies.

What we have said about design of agents so far must be embedded into the context of conducting agent experiments. We discuss this topic next.

4.4 Explaining Behavior

In placing our discussion of design of agents into the context of conducting agent experiments, we must first ask ourselves what the goal of these experiments is. The main goal of doing experiments within a synthetic approach is explaining behavior, as we have said. This can be the behavior of a natural agent, or of an artificial one. Before describing the experimental steps that need to be followed, let us highlight some core aspects of explaining behavior.

Time Perspectives for Explanations

Given that our stated goal in conducing agent experiments is to find mechanisms that underlie behavior, we can examine in more detail the kinds of explanations we are looking for. Again, what we regard as good or interesting explanations strongly depends on our research goals. Our general goal is to understand the phenomena reviewed in chapter 1. To do so, we must discuss intelligence at three different levels or time perspectives: short-term, ontogenetic,

and phylogenetic. One might add a fourth perspective concerned with what purpose a behavior serves.

1. The *short-term* perspective explains why a particular behavior is displayed by an agent based on its current internal and sensory-motor state. It is concerned with the immediate causes of behavior. We used the short-term perspective when we explained the behavior of Simon's ant on the beach. In that case, we referred to the ant's current sensory states: If stimulation on right, then turn left, and vice versa. Figure 4.14 shows how short-term explanations can be found in a robotic setup. The robot's behavior is shown in the lower right corner. Its internal state is displayed (sensors, activation levels, and weights of the neural network) in the other windows, and we can use this information to explain the behavior we are seeing. For example, we can explain why the robot has turned away from an obstacle based on its internal state, that is, the values of sensor signals, activation levels, and perhaps motor speeds. This setup has the advantage of enabling us to record anything we would like about the robot's internal state, an option we do not have for living beings like animals and humans. Clearly, if we do not have a short-term explanation of an agent's behavior, we simply do not understand how it works.
2. The *ontogenetic* perspective resorts not only to current internal and sensory-motor state but also to some events in the more distant past in order to explain current behavior. The ontogenetic perspective is also called the learning and development perspective. Explanations from the ontogenic perspective are almost universally used in the study of intelligence. The entire field of instructional sciences is based on it. When we say a student has done well on a test because he studied a lot, we reference a sequence of events in the past: the student reviewing the materials for the test repeatedly. If a robot initially crashes into obstacles but over time starts avoiding them, it has learned a behavior. Both of these explanations of the student's and the robot's behavior are framed in an ontogenic perspective.
3. The *phylogenetic* perspective asks how the behavior evolved during the history of the species. Finally, this perspective puts the agent into the context of an evolutionary process, a timescale in the very long term. An illustration of this has already been discussed: The ''peppered moth'' that changed its color from light to dark because the tree trunks had changed from light to dark as a result of industrialization.

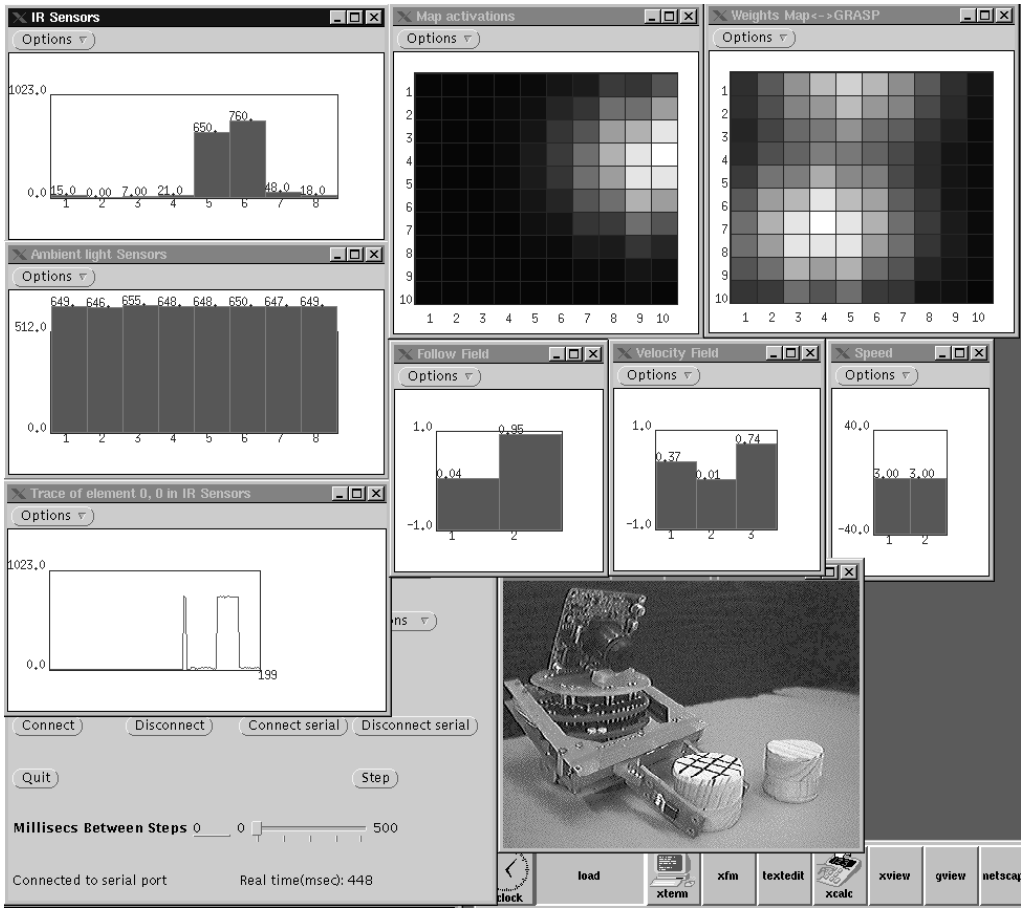


Figure 4.14 Setup for generating short-term explanations. Short-term explanations can best be made by displaying both the robot's internal state and its behavior on the screen. The robot's behavior is recorded via a video camera mounted over the experimental area. From this video information, the trajectory and other behavioral data (like the direction the robot is facing, its speed, and its direction of motion, which does not have to coincide with its direction of movement) can be extracted. The information extracted from the videotape is synchronized with the data about internal state (such as battery level, activation levels, and weights of neural networks—see chapter 5) and a time series file containing all this information is created. If this recording is performed over extended periods, behavior changes over time—that is, learning behavior—can be studied.

Throughout this book, different theoretical positions we examine attribute different weights to the three perspectives: dynamical systems place emphasis on the short-term perspective (chapter 9), connectionism and neural networks place it on the ontogenic, specifically learning (and partly development—chapter 5), and evolutionary approaches place it on the phylogenetic (chapter 8). All three kinds of explanations contribute in important ways to our understanding of intelligence. None can replace all the others.

4. One could add a fourth perspective that is not a temporal one: One can ask what a particular behavior is for; that is, how it contributes to the agent's overall fitness, a concept we elaborate on in chapter 8. In biology, this is called the *ultimate* or *functional* perspective. This question can only be answered if fitness has been defined. Except in the field of artificial evolution, this is generally not the case for autonomous agents. Moreover, in many cases, it is not obvious how a particular behavior contributes to fitness. We return to this point in chapter 8. In this book, we focus on perspectives (1), (2), and (3).

These perspectives can perhaps be best illustrated with an example. Suppose we ask why drivers stop their cars at red traffic lights. One answer would be that a specific visual stimulus, the red light, reliably leads to specific behaviors like changing gear and applying the brakes: This would be an explanation in the short term. A different answer is that individual drivers learn this rule from books, television, and driving instructors: This would be an explanation in terms of ontogenesis, learning, or development. An evolutionary explanation would deal with the historical process whereby a red light came to be used in many countries as a way of stopping traffic at road junctions. A functional explanation would be that drivers who do not stop at traffic lights are liable to have an accident, or at least be stopped by the police. (Example adapted from Martin and Bateson 1993.)

These perspectives closely resemble what is called ''the four whys'' in biology (e.g., Huxley 1942; Tinbergen 1963). What we have called the short-term perspective is also called a proximate explanation by biologists. What we have called the ontogenetic perspective is similar to its use in biology, but we have a stronger focus on learning. Our use of the phylogenetic perspective is identical to that in biology.

Table 4.5 Guidelines for conducting agent experiments. Note that this is the basic scheme and is more like a checklist rather than a step-by-step procedure.

Step	Description	Chapters
0.	Decide on research goal.	16
1.	Define the tasks/desired behaviors and the ecological niche, i.e., the task environment.	16
2.	Define the low-level specifications.	5, 16
3.	Choose a platform.	16
4.	Define the control architecture.	11–14, 16
5.	Define the concrete experimental setup and the experiments to be run.	17
6.	Before running the experiments, formulate predictions and hypotheses and provide the rationale for them. Think about how the agent's performance is to be evaluated.	17
7.	Perform the experiments; collect data about • agent behavior • internal state of the agent • sensory-motor state.	17
8.	Describe the agent's behavior and perform various kinds of statistical analyses.	4, 17
9.	Formulate explanations of the agent's behavior. Analyze the model's limitations. Report on failures.	4, 17

Conducting Experiments with Complete Agents

We have pointed out three main purposes for which one might pursue building complete agents: modeling certain aspects of natural agents, studying general principles of intelligence, and building agents for a particular task (or tasks). We have also described and compared two types of artificial, complete agents: simulated and robotic agents. In this section we summarize the guidelines to conduct scientific experiments with complete agents. An overview is provided in table 4.5. We give only a short description here; details are left for chapter 16.

Before we start conducting experiments, we have to know what *research issues* we want to investigate. Normally this should be fairly obvious: navigation behavior of desert ants, for example, or phonotactic behavior of crickets, category learning in human infants, cooperation in primate societies, or data collection on Mars. The next things to decide upon are the *tasks* or the *desired*

behaviors of the agent and its *ecological niche*. Because behavior always takes place in a particular environment, we use the term *task environment* to designate the two together. The task of the Sojourner, for example, is to collect data on the planet Mars, and its ecological niche is the surface of Mars. We have also discussed the robot cricket built by Webb (e.g., 1993). The desired behaviors of Webb's robot cricket (e.g., 1993), which we discussed earlier, are to approach a sound source from various initial positions according to principles observed in the real cricket. Then, the low-level specification needs to be defined. In other words, a decision must be made—given the agent's task and ecological niche—as to what the agent should be able to sense, what its body should look like, how it should interact with its environment, and so on. The Sojourner, for example, has to navigate on the surface of Mars while avoiding obstacles. Thus, it needs an appropriate set of sensors. On the Sojourner, cameras, bumper sensors, and proximity sensors were used to provide this ability. Its body and motor system had to be built to enable it to overcome obstacles of considerable height, which is why six wheels were incorporated, rather than four. Similar considerations apply for the robot cricket, which needs means of detecting sounds of particular wavelengths and of navigating toward the sound source.

Then, a *platform* has to be chosen; that is, how should the low-level specification be realized (implemented)? Among other things, a decision must be made whether to use simulation or a real robot. For the Soujorner, this choice was obvious: A simulation on Earth cannot produce measurements on Mars. It may not have been so obvious in the case of the robot cricket. If the designer opts for a robot, the choice is between buying a platform off the shelf or building one. The decision strongly depends on resources and know-how already available (see also chapter 16).

The next step involves defining the *control architecture*, which essentially specifies how the various parts of the low-level specification, the primitives, should be connected to produce the desired behavior. In the case of the robot cricket, this was in fact the main research issue: How can the robot cricket be ''wired up'' or programmed so that it produces a behavior comparable to the one observed in the real cricket? The control architecture—appropriately embedded in the robot cricket—thus implements hypotheses about the mechanisms underlying the real cricket's behavior. The Sojourner robot's purpose was not to understand natural intelli-

gence, but rather to achieve a particular task: Biological or psychological considerations were irrelevant. The particular control architecture chosen for an agent crucially depends on the purpose for which the robot is being designed. If the goal in building the robot is to model natural intelligence, the main considerations are biological or psychological plausibility, whereas if it is to fulfill some task, the control architecture must be chosen to implement efficient task-related behaviors.

The final step before the actual experiments can be run entails formulating *predictions (hypotheses)* about what is going to happen, given the agent's platform, control architecture, low-level specification, ecological niche, and task. In addition, decisions about the *evaluation* of the robot's performance have to be made. It is not effective research design simply to run a large number of experiments, collect data, then think about evaluation at the very end. One should be clear before any experiments are run about what types of data one wants to collect and how one wants to analyze that data, for example, in terms of statistical analyses. Of course, this can be an iterative process whereby preliminary experiments reveal what kinds of data are most relevant, but as a general rule of thumb, it is good practice thinking about these issues beforehand. The case of the Sojourner robot makes the point very clearly: Imagine what would have happened if evaluation criteria had been derived only after the robot had been sent to Mars! The same case could be made about the robot cricket. In any case, from a purely scientific perspective, hypotheses always have to be formulated before the experiments are actually performed.

When running the actual experiments you need to *collect data* about all relevant aspects of the robot's behavior. This includes the behavior as seen by an outside observer and the robot's internal state, for example, the sensor data and data on the neural network dynamics and motor states. The setup from figure 4.14 can be used to record behavior and internal states automatically. Finally, once you have all the data you need, you can start describing the robot's behavior and analyzing these data. There are many ways to describe behavior, and the descriptions can be made on very different levels. For example, we can give verbal descriptions, or we can draw the trajectories exhibited, preferably automatically. We can approach this more quantitatively and do various kinds of statistical analyses. Statistics often lend themselves most readily to interpretation if they are represented graphically. We can also

describe agent behavior in terms of mathematical models (e.g., differential equations). Additional methods can be found in any textbook on general experimental methodology. Note that a description of behavior implies a segmentation of behavior: The behavior has to be cut up into meaningful pieces, the segments, to be described effectively. For example, saying that someone is eating, drinking, getting up, and leaving the table represents a segmentation of the behavior ''eating dinner.'' In the robotic domain, examples of behavior segments would be turning toward a light source, picking up a peg, following a wall, or recharging its batteries. In addition to a description of its behavior, the robot's performance needs to be evaluated. Experiments can be evaluated in many ways. We leave providing a detailed overview to chapter 17 but present examples of experiment evaluation as we go along.

Issues to Think About

Issue 4.1: Hybrid Specifications—Choice of Tasks

Earlier in the chapter, we made a preliminary try at designing a robot to serve tea in a restaurant. On the one hand, such a robot has to know about its environment: about restaurants, objects, and procedures in the restaurant. On the other, it has to act physically in the restaurant: It must actually bring the tea to customers. We argued that if you start by designing the high-level ontology and the low-level specification using a design in which concepts emerge through agent-environment interaction, there will be incompatibilities between the two. If that is so, how *do* we design a tea-serving robot in a principled way? We honestly don't know. You will find, as you read through this book, that the kinds of behaviors we can engender through emergent designs are, though interesting, not sufficient to produce such complex behaviors as those required in a restaurant. The object recognition problems are enormous, the object manipulation skills, considerable. Just think of preparing a cup of tea, putting it on a small tray, and carrying it to the—right—customer. It is also implied that the robot would need some way of communicating with the customers. We could probably produce a ''hack'': We could try to introduce physical constraints in the environment: For example, we could specify that the cups are always found in exactly the same location, we could put identifiers

on the tables and the different kinds of tea, and we could arrange for smooth grounds so the robot could use wheels. We could also scale down the robot's task by not having it manipulate the tea cups themselves: Personnel could put them on the robot, and the customers could pick them up themselves once the robot has arrived at their tables. But what would we then learn about the principles of intelligence? Presumably not too much. So the conclusion seems to be that it may be premature to actually try to build a tea serving robot. But we might be able to make a compromise. We could make some simplifications, changes to the environment, and try to cause at least some of the behaviors to emerge. If these changes were done right, the robot might actually be able to learn to look out for cluttered tables with no customers, for example, and recharge its battery on its own, if required. Alas, this kind of study has not been widely attempted. One project that moves in this direction, however, is the sewage system robot project that we outline in chapter 18. As we see later on, the choice of appropriate tasks is crucial to the success of an agent experiment. Try to apply these considerations to an application of your choice.

Issue 4.2: Limitation by Low-Level Specification

We have stressed the limitations imposed by high-level ontologies. But robots, and humans for that matter, are also constrained. Anything for which our sensory system makes no provision, we can simply not sense. Our visual system can detect electromagnetic waves only within a certain limited range. Anything outside is simply not accessible to the visual system. (We function as well as we do because of the redundancy built into our sensory system that enables us to detect events beyond the capacity of a sensor. For example, although our eyes can't measure temperature, we can often ''see'' whether objects are really hot or really cold.) Try to think of other limitations of our sensory system to get an idea of what our own ''low-level specification'' is and how it constrains our potential interactions with the real world.

Points to Remember

- The agents of highest interest for our purposes are complete agents. They are autonomous, self-sufficient, embodied, and situated. They have been given the name ''Fungus Eaters.''

- Self-sufficient agents can perform multiple tasks, can exhibit multiple behaviors in the real world over extended periods of time; that is, they do not incur an irrecoverable deficit in any of their resources. Self-sufficiency implies adaptivity.
- Self-sufficiency always pertains to a particular ecological niche. An ecological niche is the range of environmental variables within which a species, or an autonomous agent, can exist. Agents are always designed (by an engineer or by evolution) for a particular ecological niche: There is no universal agent in the real world. If the specific properties of the ecological niche are exploited, scalability of learning algorithms can often be achieved.
- Because self-sufficient agents always have many tasks, they have to solve the problem of behavior control: loosely speaking, the problem of doing the right thing at the right time. Action selection designates the problem of choosing an action in a particular situation from a given set of actions. The problem with the action-selection approach is that in general there is no straightforward mapping of desired behaviors to internal actions.
- Autonomy means independence of control. This characterization implies that autonomy is a property of the relation between two agents, in the case of robotics, of the relations between the designer and the autonomous robot. Self-sufficiency, situatedness, learning or development, and evolution increase an agent's degree of autonomy.
- A situated agent acquires all information about the environment from its own perspective through its sensory system.
- Embodiment means existing as a physical entity in the real world, that is, as a robot. Embodied agents can also be simulated, as is often done in virtual reality environments. The positioning of the sensors on the agent must be specified because where the sensors are positioned affects system-environment interaction. Moreover, how the control architecture is embedded in the agent must also be defined.
- There are four kinds of adaptation: evolutionary, physiological, sensory, and adaptation by learning. All operate on different timescales.
- There are three potential goals when building an artificial agent: (1) building an agent for a particular task or a set of tasks, (2) studying general principles of intelligence, and (3) modeling certain aspects of natural systems, that is, humans or animals.

- Standard simulations differ from agent simulations in that a simulated agent interacts with a simulated environment through its own sensory-motor system, whereas in a standard simulation, the agent does not interact with the environment at all.
- The frame-of-reference problem conceptualizes the relation between the designer, the observed agent, the artifact to be designed, and the environment. There are three issues: perspective, behavior-versus-mechanism, and complexity.
- A high-level domain ontology is a systematic account of the basic components, the primitives, that will be used in the system. Anything the system will be able to do builds on this ontology. This holds also for the communication with the environment.
- A low-level specification is the equivalent of a domain ontology for a robot: It includes the body, the sensory and motor systems, and potential connections. Robots should be specified in terms of low-level specifications rather than high-level ontologies. Hybrid specifications should be avoided.
- Sensor spaces typically have very large numbers of states. To make them manageable, we need to exploit constraints that we get from interaction with the environment.
- The term ''emergence'' is used primarily in three different ways: (1) something surprising and not fully understood, (2) a property of a system not contained in any one of its parts, and (3) behavior that is not preprogrammed that arises from agent-environment interaction. Definition (2) is the meaning intended in the self-organization and artificial life communities, and (3) is the one in the autonomous agents field. Our goal in building autonomous agents is to design for emergence.
- When conducting agent experiments, the following steps must be taken (though not necessarily in this order): decide on research goal; define tasks or desired behaviors and ecological niche; define low-level specifications; define control architecture; choose a platform; define concrete experimental setup and experiments to be run; formulate predictions; run experiments and collect data; describe agent's behavior; and formulate explanations.

Further Reading

Hendriks-Jansen, H. (1996). *Catching ourselves in the act: Situated activity, interactive emergence, evolution, and human thought*. Cambridge, MA: MIT Press (A Bradford Book). (A philosophical treatment of the entire field of embodied cognitive science. The idea of emergence is given extensive treatment. Contains many examples

described in great detail. Also discusses the major criticisms not only of classical approaches in AI but also in ethology and psychology.)

McFarland, D. (1995). Autonomy and self-sufficiency in robots. In L. Steels and R. Brooks (Eds.), *The artificial life route to artificial intelligence: Building embodied, situated agents* (pp. 287–309). Hillsdale, NJ: Lawrence Erlbaum. (A comprehensive discussion of the concepts of autonomy and self-sufficiency in the context of the behavioral economics approach.)

Toda, M. (1982). *Man, robot, and society*. The Hague The Netherlands: Nijhoff. (An entertaining and intelligent discussion of many fundamental problems in cognitive psychology and the psychology of emotion. Masanao Toda is one of the leading psychologists in Japan who for many years has been working on developing a comprehesive theory of emotions.)