



Chapter 7

The Subsumption Architecture

7.0 CHAPTER OVERVIEW

While the most specific aim of this chapter is to provide a program to be used to control the LEGO Tortoise that was constructed in Chapter 6, it has a more general aim as well: to provide a brief introduction to the subsumption architecture (Brooks, 1989, 1999, 2002), which is a successful and flexible approach that has been used to program a number of different robots. The subsumption architecture explicitly rejects the classical view that cognition's purpose is planning, and that cognition mediates perception and action (see our earlier discussion of this issue in Chapter 3). Instead, the subsumption architecture views a robot as being a stacked set of modules. Each module is a sense-act mechanism – thinking or representation is removed as much as possible. Lower levels in a subsumption architecture are modules that govern more basic or general abilities. Higher levels in a subsumption architecture are modules that take advantage of the abilities provided by lower levels and provide more complex abilities. Higher levels might provide weak control over lower levels (for instance, by inhibiting them for a moment); lower levels have no access to, or control of, modules that are higher in the hierarchy.

This chapter proceeds by describing some of the motivation for the development of the subsumption architecture, with a brief account of a famous example (Brooks, 1989). It then provides a more detailed example of this type of architecture by providing the code for the various levels that are used to control the behaviour of the LEGO Tortoise. At the end, it provides a brief description of the behaviour of both the

complete Tortoise and the Tortoise when higher-level modules are removed. The implications of such behaviour are discussed at the end of the chapter.

The subsumption architecture is an extremely useful approach for developing our robots, particularly as they become more complicated because of employing multiple sense-act mechanisms. Additional examples of the subsumption architecture are also provided for the Lemming robot described in Chapter 8 and for the antiSLAM robot that is the topic of Chapter 9.

7.1 A SANDWICH OF VERTICAL MODULES

7.1.1 Cognitivism

The cognitivist movement arose in psychology in the late 1950s as a reaction against the behaviourist school (Gardner, 1984). A growing number of psychologists were frustrated with behaviourism's reluctance to explore internal processes that could not be directly observed, as well as with its general position that humans were passive responders to environmental stimuli.

The cognitive revolution responded to behaviourism by adopting an information-processing metaphor (Miller, 2003). Rather than passively responding to the environment, humans were described as active processors who received information, manipulated this information to make important content explicit, and then used this explicit content to guide action. This general approach has been called the *sense-think-act cycle* (Pfeifer & Scheier, 1999). The hypothesis that cognition is information processing amounts to the claim that the human mind is a complex system that receives, stores, retrieves, transforms, and transmits information (Stillings et al., 1987). For cognitivists, explanations must include accounts of the processes that manipulate information. As a result, successful cognitive theories must include proposals about the manner in which information is represented, as well as proposals about the rules or procedures that can transform these representations (Dawson, 1998). These theories are instances of what has come to known as the classical approach, as the symbolic approach, or as the representational theory of mind (Chomsky, 1980; Fodor, 1968, 1975; Newell, 1980; Pylyshyn, 1984).

7.1.2 The Classical Sandwich

There are many important consequences of adopting the classical approach. Two of these involve the relationship between perception, cognition, and action (Hurley, 2001). First, cognition – the rule-governed

manipulation of representations — is viewed as being the central characteristic of the mind. Second, perception and action are seen as being peripheral characteristics of the mind, as well as being separate from one another.

These two consequences arise because the classical view tacitly assumes that one of the primary purposes of cognition is planning (see Chapter 3). For example, it has been argued that mind emerged from the natural selection of abilities to reason about the consequences of hypothetical actions (Popper, 1978). This permitted fatally incorrect actions to be discarded before actually being performed. “While an uncritical animal may be eliminated altogether with its dogmatically held hypotheses, we may *formulate* our hypotheses, and criticize them. Let our conjectures, our theories die in our stead!” (p. 354).

Classical theories tend to take on a stereotypical form when it is assumed that the function of cognition is planning. Hurley (2001) described this form as a set of vertical modules that stand between (or that are sandwiched by) perception and action. Each vertical module is classical in nature — each involves a particular form of representation, and particular processes that modify these representations. Critically, there are no direct connections between perception and action (see Figure 3-1 in Section 3.3). That is, perception can only indirectly inform action, by sending information to be processed by the central, vertical modules, which in turn ultimately choose which action is to be performed. In her critique, Hurley called this structure the *classical sandwich*.

Many researchers now question the classical sandwich, and are considering alternative roles for cognition. Some have argued that cognition is not used to plan, but is instead used to control action (Clark, 1997; Varela et al., 1991). The classical sandwich is being disassembled, because direct links between perception and action are appearing in cognitive theories (Brooks, 1999, 2002). We earlier saw an example of this with the “leaky mind” model depicted in Figure 3-3 (Section 3.12).

7.2 THE NEW LOOK AND ITS PROBLEMS

7.2.1 The New Look in Perception

Why would the classical sandwich be challenged? One answer to this question comes from exploring an example of perceptual theory and its performance when it is imported into a behaving robot.

In the late 1940s and early 1950s, psychologist Jerome Bruner and his colleagues performed a number of experiments that led to a radically

new, cognitive theory of perception (Bruner, Postman, & Rodrigues, 1951). These experiments indicated that subjects' perceptual experiences were strongly influenced by their expectations or experiences.

The implication of such results was that perception (i.e., categorizing the visual world, as opposed to sensing light) is equivalent to cognition (Bruner, 1957). "A theory of perception, we assert, needs a mechanism capable of inference and categorizing as much as one is needed in a theory of cognition" (Bruner, 1957, p. 124). The view that perception was in essence an active cognitive process became known as the *New Look*. More modern variants of this perspective have also appeared (Gregory, 1970; Rock, 1983).

For Bruner (1957), the New Look was "a general view of perception that depends upon the construction of a set of organized categories in terms of which stimulus inputs may be sorted, given identity, and given more elaborated, connotative meaning" (p. 148). This is consistent with Hurley's (2001) classical sandwich, because "perception" would merely deliver the stimulus inputs, and perceptual categorization would be accomplished by the vertical modules.

7.2.2 Shakey Implications

Beginning in 1966, the Stanford Research Institute conducted research on a robot nicknamed "Shakey" (Nilsson, 1984). Shakey plotted its own path through a controlled indoor environment, using a television camera, an optical range finder, and touch sensors. Shakey communicated what it sensed via radio signals to a central computer that updated Shakey's model of the world, and planned Shakey's next behaviour. Shakey illustrates the classical sandwich, with its vertical modules being contained in the central computer.

Shakey's model of the world was a set of predicate calculus expressions. This predicate calculus also represented Shakey's goals. A planning system (called STRIPS) would attempt to derive a sequence of actions to convert the current model to one in which the goal was accomplished. Shakey would then execute the sequence of actions to physically accomplish the goal.

Shakey was capable of performing a number of impressive tasks, many of which are illustrated in the film *SHAKEY: Experimentation in Robot Learning and Planning*, which is available from <http://www.ai.sri.com/movies/Shakey.ram>. It could plan routes through its environment, navigate around obstacles, and move obstacles (large painted blocks) to desired locations. However, Shakey also revealed two of the main problems with the classical sandwich.

First, Shakey was as successful as it was because it was placed in a carefully constructed and controlled environment that simplified visual processing and model building. “Shakey only worked because of a very careful engineering of the environment. Twenty years later, no mobile robot has been demonstrated matching all aspects of Shakey’s performance in a more general environment” (Brooks, 1999, p. 61).

Second, Shakey was extremely slow. Even when placed in a tailored environment that was not complicated to sense or to model, it took several hours for the robot to complete a task (Moravec, 1999). This was because maintaining and using the world model was computationally expensive. The problem with the sense–think–act cycle in robots like Shakey is that by the time the (slow) thinking is finished, the resulting plan may fail because the world has changed in the meantime. This problem is dramatically accentuated by increasing the complexity of the world model that is maintained in the interior of the classical sandwich.

7.3 HORIZONTAL LAYERS IN THE HUMAN BRAIN

7.3.1 Evidence from Action

The classical sandwich amounts to the claim that sensing is separated from acting by a great deal of thinking, modelling, and planning. However, this claim begins to be severely challenged when researchers study the neural mechanisms that coordinate perception and action (see also Section 3.7.2).

For example, consider the examination of the patient known as DF, who suffered extensive brain damage as the result of carbon monoxide poisoning (Goodale et al., 1991). DF’s brain injuries did not impair basic sensation, such as detection of colour, or the spatial resolution of images. However, higher-level perception was severely impaired. DF had severe visual form agnosia, and could not describe the orientation or shape of any visual contour, no matter what visual information was used to create it.

Amazingly, while DF could not consciously report orientation or shape information, such information was available to control some of her behaviour. In particular, when DF was asked to perform a motor activity, such as grasping an object, or inserting an object through an oriented slot, her actions were identical to controls, even to the fine details that are observed when such actions are initiated and then carried out. “At some level in normal brains the visual processing underlying ‘conscious’ perceptual judgments must operate separately from that underlying the ‘automatic’ visuomotor guidance of skilled actions of the hand and limb” (Goodale et al., 1991, p. 155).

DF's brain injuries caused visual form agnosia, but left visuomotor coordination intact. Importantly, other kinds of brain damage produce a very different pattern of deficits that support the notion of "separate operation" noted above.

Patients who suffer damage to the posterior parietal cortex can exhibit optic ataxia, in which they are unable to use visual information to reach out and grasp objects when presented in the part of the visual field affected by the brain injury. Some of the motor skills studied in patient DF have also been studied in the patient VK, who was suffering optic ataxia (Jakobson, Archibald, Carey, & Goodale, 1991). One of the main differences between VK and DF was that VK demonstrated a number of visuomotor abnormalities. For instance, the size and shape of her grasp were only weakly related to the size and the shape of the to-be-grasped object, and grasping movements took much longer to be initiated and to be executed. A second main difference was that VK, unlike DF, had no difficulty recognizing the orientation and shapes of visual contours.

In short, DF and VK illustrate the double dissociation between brain mechanisms responsible for the conscious awareness of visual form and brain mechanisms responsible for complex visually guided actions, such as grasping objects.

7.3.2 Sandwich Alternative

These results can be used to argue that the human brain is not completely structured as a "classical sandwich." On the one hand, Goodale concedes that one likely function of the visual system is the creation of a model of the external world; this is the kind of function that the classical sandwich captures, and is disrupted in DF (Goodale & Humphrey, 1998). However, a second function, revealed by the double dissociation, is the control of action. This is accomplished by converting visual information directly into motor commands. This is not part of the classical sandwich, because it assumes that there is a much more direct link between vision and action.

It has been argued that the two functions mentioned above can co-exist, can interact, and can complement one another (Goodale & Humphrey, 1998). However, some would argue that the typical relationship between vision and action has much more to do with controlling action than building models. This has led to a proposed architecture that is a direct challenge to the classical architecture and to the sense-think-act cycle. We now turn to considering this alternative proposal.

7.4 HORIZONTAL LINKS BETWEEN SENSE AND ACTION

7.4.1 A Sandwich Alternative

One alternative to Hurley's (2001) classical sandwich is much more in line with the results from neuroscience that indicate that 1) an important function of vision is the control of action, and 2) that this function is mediated by pathways that are distinct from those involved in constructing models or representations of the world (Goodale, 1988, 1990, 1995; Goodale & Humphrey, 1998; Goodale et al., 1991; Jakobson et al., 1991). The alternative is to replace the classical sandwich's vertical layers that separate perception from action with horizontal layers that directly connect perception and action.

One of the primary motivations for such horizontal layers is to recast what cognition is all about. The classical approach, as illustrated with Shakey, is that cognition amounts to planning, and that this planning is required to mediate perception and action. In short form, the classical view endorses the sense–think–act cycle. The alternative view is to assume that cognition is not planning, but instead is the control of action (Clark, 1997). “The idea here is that the brain should not be seen as primarily a locus of inner *descriptions* of external states of affairs; rather, it should be seen as a locus of internal *structures* that act as operators upon the world via their role in determining actions” (p. 47). Importantly, these structures serve as links between sensing and acting, not as general processes that stand between sensing and acting.

This alternative view represents a strong reaction against the notion of central control that is fundamental to classical cognitivism. In the classical sandwich, each vertical layer is defined by a particular representational medium (i.e., symbols of a particular type) and by a set of rules that manipulate these representations. In addition, though, there must be some control mechanism that chooses which rule to apply to the symbols at any given time. In the thinking and problem-solving literature, control is usually described as deciding “what to do next,” or as searching a problem space (Simon, 1969).

Much of cognitive science is inspired by the metaphor of the digital computer (Pylyshyn, 1979), or Turing's universal machine (Turing, 1936, 1950). In these devices, control is centralized, and is used to choose one rule at a time to manipulate symbols. Not surprisingly, classical cognitive theories usually appeal (either directly or indirectly) to some sort of central control mechanism.

Many of the reactions against classical cognitivism, such as the connectionist movement (Schneider, 1987) or situated cognitive science (Greeno & Moore, 1993; Touretzky & Pomerleau, 1994), point to problems with the

classical view that are due to this central control. Classical systems are often described as slow, brittle, and unable to gracefully degrade (Feldman & Ballard, 1982). One solution to such problems is to decentralize control.

Historically, the first major modern proposal for decentralizing control is to take advantage of the distributed representations that are characteristic of artificial neural networks (McClelland & Rumelhart, 1986; Rumelhart & McClelland, 1986). Arising slightly later, and having a more recent impact, are the sense-act models that have been developed within behaviour-based robotics (Brooks, 1999, 2002; Pfeifer & Scheier, 1999). We will consider one of these approaches, Brooks' subsumption architecture, in more detail in following sections.

However, it is important to be aware that the possibility of decentralized control has broader implications for cognitive science. For example, arguments against the "Cartesian theatre" view of consciousness (Dennett, 1991, 2005) are completely consistent with this sort of decentralization. So too is the notion of high-level cognitive phenomena emerging from a "society of mind" (Minsky, 1985, 2006). Similarly, decentralized control makes possible the notion of the mind leaking into the environment, challenging the classical views of boundaries of the mind (Clark, 1997, 1999; Wilson, 2004).

7.5 THE SUBSUMPTION ARCHITECTURE

7.5.1 Modularity of Mind

One of the key innovations in modern cognitive science is the idea that many cognitive functions are modular (Fodor, 1983). A module can be viewed as a special purpose machine that only has access to a limited amount or type of information, which permits the machine to be fast. The module is designed to solve a particular problem, is associated with specific neural circuitry, and cannot be influenced by the contents of higher-order beliefs, desires, or expectations. A module would not be part of Bruner's (1957) New Look!

Modularity is not inconsistent with the classical sandwich. Many of the horizontal layers that separate perception from action could be modular in Fodor's (1983) sense. For example, the computational theory of vision proposed by David Marr (Marr, 1976, 1982; Marr & Hildreth, 1980; Marr & Ullman, 1981) consists of a set of modules that produce a sequence of preliminary representations of visual information (the raw primal sketch, the full primal sketch, the 2½D sketch). These modules are used as the foundation for a meaningful, useful mental representation of the world. Marr's theory is modular, and is also classical in exactly the sense that is questioned by Hurley (2001).

7.5.2 Vertical Modules

The subsumption architecture that has been proposed by roboticist Rodney Brooks (1999) is modular, but explicitly departs from the classical sandwich, and rejects the sense–think–act cycle. The subsumption architecture is a set of modules, each of which can be described as a sense–act mechanism. That is, every module can have access to sensed information, as well as to actuators. This means that modules in the subsumption architecture do not separate perception from action. Instead, each module is used to control some action on the basis of sensed information.

A second characteristic of the subsumption architecture is a hierarchical arrangement of modules into different levels. Lower levels are modules that provide more basic or more fundamental sense–act functions. Higher levels provide more complex sense–act functions, which depend upon (or take advantage of) those provided by lower-level functioning.

The hierarchical structure of the subsumption architecture reflects the generality of the sense–act function provided by each level. The functions provided by lower levels are more general, in the sense that they are required in a very broad array of situations. Higher-level functions are more specific, designed to be used in a narrower range of situations.

The hierarchical relationship amongst levels in the subsumption architecture is also reflected in how they communicate with one another. Consider the lowest level, Level 0, and a level built immediately above it, Level 1. Level 1 has access to the sense data of Level 0, and can send signals that alter Level 0 (e.g., by inhibition). However, the reverse is not true: Level 0 cannot access or influence Level 1. In other words, the hierarchy of the subsumption architecture is one of control.

Importantly, control in the subsumption architecture is not centralized. There is no central clock, and no serial processing. All of the levels in the architecture run in parallel and asynchronously.

The modular and hierarchical nature of the subsumption architecture is also reflected in how it is created. A designer decides on what the lowest level, the broadest sense–act function, should be. This level is created, and then never revisited. “We start by building a complete robot control system which achieves level 0 competence. It is debugged thoroughly. We never alter that system” (Brooks, 1999, p.10). Then this process is repeated for the next level, and continues until all of the levels in the subsumption architecture have been completed.

7.6 ADVANTAGES OF THE SUBSUMPTION ARCHITECTURE

7.6.1 Reasons for Revolution

Brooks (e.g., 1999) proposed the subsumption architecture as an explicit reaction against classical notions of sense–think–act and centralized control, particularly as these notions were realized in classical research on autonomous robots. Not surprisingly, Brooks has argued that the success of his own robots illustrates that the subsumption architecture has many demonstrable advantages over robots that are designed with the classical sandwich in mind.

7.6.2 Coping with Multiple Goals

Grey Walter’s robots (Grey Walter, 1950a, 1950b) had to accomplish more than one goal: seeking moderate light, approaching such light when it was found, avoiding bright light, and avoiding physical obstacles.

A classical system that uses centralized control, such as Shakey, must carefully consider all of the robot’s goals at any given time to plan an appropriate course of action. This process becomes more and more complicated as the number of goals multiplies. Furthermore, as changes in the environment occur, their impact on the robot’s goals must be re-evaluated, leading to what has been called the frame problem (Pylyshyn, 1987). In the frame problem, a classical system is lost in thought, evaluating the impact of world changes on its world model, unable to perform actions in a timely manner.

Brooks (e.g., 1999) argued that the subsumption architecture can deal with multiple goals that are in conflict or change in priority. This is because different goals are associated with different levels, and each level pursues these goals in parallel. Coping with multiple goals emerges from the control structure of the architecture. For example, a higher level pursuing the goal of avoiding objects can inhibit lower levels pursuing the goal of moving, but only when an object is encountered to make the higher level’s goal a more immediate priority.

7.6.3 Combining Multiple Sensors

A classical system is also challenged when the number and types of sensors used to create a model increase. This leads to a need to increase the complexity of the model, and the resources used to update the model, which is a variation of what is known as the packing problem (Ballard, 1986).

The subsumption architecture provides an elegant approach to dealing with multiple sensors, because usually each level uses only a subset of the sensors that are available. As well, increases in computational demands

are kept in check because each level is converting sensed information into action, and is not updating an internal model of the world.

7.6.4 Robustness

A robust robot generates useful behaviour even when it has problems with sensors. The subsumption architecture is argued to be robust (Brooks, 1999) because of its hierarchical nature. Lower levels provide the most basic and the most generally applicable behaviours, while higher levels provide more sophisticated and specialized capabilities. If higher levels in the architecture fail, or are processing inputs too slowly, the lower levels are still operating. As a result, the robot still performs some appropriate behaviour under challenging situations.

7.6.5 Speed with No Modelling

One of the key criticisms of the classical approach is that the need to maintain an internal model of the world results in an agent that is too slow to take action under the real-time demands of the world. The subsumption architecture deals with this problem by removing the need to build internal models. Instead, the world is used as a model of itself, which the subsumption architecture senses but does not represent. “The realization was that the so-called central systems of intelligence – or core AI as it has been referred to more recently – was perhaps an unnecessary illusion, and that all the power of intelligence arose from the coupling of perception and actuation mechanisms” (Brooks, 1999, p. viii).

7.7 CONCRETE EXAMPLES

7.7.1 Walking Robots

The advantages of the subsumption architecture have been demonstrated in a number of different walking robots. One famous example is Genghis, a six-legged walking robot (Brooks, 1989). Each leg is affected by two motors, one for swinging it back or forth, and another for lifting it up or down. The subsumption architecture controls leg movement, and interesting walking behaviour emerges from interactions amongst the architecture’s layers.

Level 0 for this robot is **standup**; each leg’s motors are set to hold the leg in a position so that all legs together enable the robot to stand.

Level 1 is **simple walk**. This includes mechanisms to set a leg down if it is not down already, to balance the robot (so that if one leg moves forward, the other legs will move backward slightly, and to move legs up and forward. Most of these mechanisms work independently for each leg, but one mechanism totals the back-and-forth position of each leg in

an attempt to coordinate the legs. The result is that different walking gaits can be produced in the robot.

Level 2 is **force balancing**. The force on each leg is monitored by measuring the force placed on the motor that raises a leg up or down. Whenever the force is too high, a signal is sent to lift the leg. This is an attempt to compensate for rough terrain.

Level 3 is **leg lifting**. By measuring the force on the motor used to swing a leg forward, this level determines if the leg is hitting an obstacle, and will send a signal to lift the leg higher if the measured force is too high.

Level 4 is **whiskers**. Two whiskers on the front of the robot are used to detect obstacles. If a whisker is depressed, the front leg of the robot on the whisker's side will be raised higher.

Level 5 is **pitch stabilization**, and is used to provide more sophisticated balancing than is provided by Level 2. Pitch stabilization senses the angle of the robot's body, and will lift either the front or the rear legs to provide better stability.

Level 6 is **prowling**. Six infrared sensors are mounted on the front of the robot, and with this level the robot will only move if it has detected something else that has moved nearby.

Level 7 is **steered prowling**. The infrared sensors are capable of noting the direction of detected movement, and this level sends signals to move legs in such a way that the robot turns in this direction.

This subsumption architecture was built into Genghis, and it produced a number of very interesting behaviours. The robot could walk over a number of different terrain types without having to be altered from one terrain to the next. It could demonstrate more than one gait, including one in which subsets of legs supported the robot as alternating tripods, and one in which the gait ripples through the legs from the back to the front. The robot could follow moving objects, such as people, using its higher architectural levels.

7.7.2 The Tortoise

The subsumption architecture for Genghis was wired into its structure. The basis for a sense-act relation in a level was a simple machine called an augmented finite state machine, which is a simple device that has a set of registers for storing information such as machine states, a finite state machine that determines what should happen to registers given the current machine state, and wires that permit one of these machines to send signals to another. Fifty-seven such machines were wired into a network to provide Genghis' built-in subsumption architecture (Brooks, 1989).

In the pages that follow, we will explore our own subsumption architecture for the LEGO Tortoise. Instead of soldering one together, we will attempt to use the spirit of the subsumption architecture to guide our writing of tasks in NXC. The complete code for this robot is available from the website that supports this book (<http://www.bcp.psych.ualberta.ca/~mike/BricksToBrains/>).

7.8 LEVEL 0, BASIC MOVEMENT

7.8.1 A Fundamental Function

To begin the Tortoise program, we must first decide on the most fundamental function to include as the lowest level of a subsumption architecture. Our choice for Level 0 is *basic movement*.

By basic movement, we mean a functional layer that will cause the Tortoise to move itself forward. The function is so basic that it has no sensors. However, it does have direct access to the drive motor. The purpose of this function is to ensure that the drive motor is running, causing the front wheel to move the robot.

The NXC code for Level 0 is provided below. Note that this level is defined by a single task, called `task level_0 ()`. This task turns on the motor that drives the front wheel (which is given the name `DriveMotor` in the main task), and propels it forward at `DriveSpeed` (which is also defined in the main task). As long as this task runs, this motor is on, because the command that turns the motor on in a forward direction (`OnFwd`) is contained within a `while(true)` loop.

If the Tortoise only used Level 0, then how would it behave? It would move in one direction, and its motor would not stop. It might move in a straight line, or it might turn; this would depend completely upon the direction that the front wheel was pointing when Level 0 started. Whatever direction it moved, this direction would not change, because Level 0 has no influence on the steering motor.

```
//Level 0: Turn the drive motor on.
int DriveSpeed;
task level_0(){
    while(true){
        OnFwd(DriveMotor, DriveSpeed);
    }
}
```

7.9 LEVEL 1, STEERING

7.9.1 Exploration

The logic of the subsumption architecture is that higher-level layers that take advantage of whatever functions are already provided by lower layers. For the Tortoise, Level 1 is **steering**. This level causes the steering motor to turn on, resulting in the front wheel turning. It does not cause the front wheel to drive the robot, which is instead accomplished by the lower Level 0.

This layer operates the steering motor at the front of the Tortoise. It is slightly more complicated than Level 0, because the front motor can be turned at different speeds: a medium speed for exploration, a fast speed when the robot is dazzled, and at zero speed—the motor is off—when the robot has sensed medium light.

The NXC code for Level 1 (`task level_1 ()`) is provided below. Note that this code is written to reflect the fact that later, higher levels in our architecture will affect robot turning. They will do this by sending a signal down to this level that sets the speed at which the turning motor will run.

When Level 1 is considered not by itself, but instead in the context of the pre-existing Level 0, we can see how it takes advantage of basic movement to advance robot behaviour. That is, when both of these layers are operating, the robot will move forward and change direction; when it is exploring or avoiding it produces a distinctive “staggering” motion. This is only possible because the steering level is taking advantage of the basic movement provided by Level 0.

```
//Level 1: Turn the steering motor on.
int TurnSpeed;
task level_1(){
    while(true){
        OnRevReg(TurnMotor, TurnSpeed, OUT_REGMODE_SPEED);
    }
}
```

7.10 LEVEL 2, SENSING AMBIENT LIGHT

7.10.1 Light Affects Lower Levels

The next level in the Tortoise is Level 2, which brings the phototropisms to life. Recall that Grey Walter’s robots explored in the dark, moved straight in moderate light, and then turned away when they were dazzled by bright light. The NXC code below for `task level_2 ()` shows how such behaviour is added to the Tortoise.

Level 2 is an infinite loop that always measures light by reading the `LightSensor`, which is set to RAW mode by the main task. This means that it returns a large number in the dark, and a smaller number when brighter light is detected.

The Level 2 task compares the current light sensor reading to two threshold values that are defined in the main task, called `dark` and `bright`. If it is dark, nothing is done – the robot explores in the fashion defined by Levels 0, 1; because steering is initiated, the pilot light is turned on. Note that during exploration, the robot is driven at half speed. If moderate light is detected, then the front turning motor is stopped by sending a signal to the Level 1 task, the pilot light is extinguished, and the robot is driven at full speed. If bright light is detected, the turning motor is run at half speed, the pilot light is turned on, and the robot is driven at full speed.

Note that this level works by using sensed light to change the behaviours of lower levels. The signals that are sent depend upon the amount of light sensed, and manipulate the lower levels in such a way that the general light-sensitive behaviours that Grey Walter described are produced. Note, too, that what this level senses is affected by the operations of lower levels, which position the robot – and its periscope mirror – in particular positions relative to whatever light sources might be in the environment.

```
task level_2(){
    while(true){
        OnFwd(PilotLight, lightSwitch(TurnSpeed)); //Pilot lights on if turning.
        if (Vision == 1) {See = Eye;}
        //Sensor in dark threshold
        if (See <= dark){
            DriveSpeed = HalfDrive;
            TurnSpeed = FullTurn;
        }
        else {
            //Sensor in moderate threshold
            if (See < bright){
                DriveSpeed = FullDrive;
                TurnSpeed = Zero;
            }
            else {
                //Sensor in bright threshold
                DriveSpeed = FullDrive;
```

```

        TurnSpeed = HalfTurn;
    }
}
}
}
//This just toggles the lights.
int lightSwitch(x){
    if (x == 0) return 0;
    else return 100;
}

```

7.11 LEVEL 3, OBSTACLE AVOIDANCE

7.11.1 Sophistication from Tweaking

With the first three levels working, the Tortoise will explore the environment by sensing ambient light. In many instances, this exploration will cause it to bump into an obstacle. Level 3 is **obstacle avoidance**, which permits the Tortoise to find its way around an obstacle when it is encountered. An obstacle is detected when the robot's shell is depressed, and one or more of the touch sensors are triggered. This level links obstacle avoiding behaviour to this situation. Importantly, the ability of Level 3 to do this depends upon the behaviours created by all of the lower levels.

Level 3 operates by constantly reading the touch sensors in front, which is done by the `Bumpers` expression in the NXC code below (`task lvl3()`). If the sensors return a value of 1, then an obstacle has been encountered, and it is depressing the shell. The Level 3 task then “tweaks” the robot's behaviour in an attempt to move away from the obstacle. These “tweaks” consist of sending signals that are used by, and change the behaviour controlled by, the lower levels.

If the shell is depressed, the first thing that Level 3 does is make the robot's turning insensitive to light by setting `See = 0`, which affects Level 2. It then sets the sensed light to dark (overriding the light sensor) for a short period of time, and then sets it to bright, again for a short period of time. The duration of these states depends on current light conditions. These states are toggled back and forth until the shell is no longer depressed. At that time, the light sensor is reactivated, and all routines – including Level 3 – go back to their usual operation.

By toggling the two sensed light conditions, Level 3 sends signals that affect the behaviour of Level 2. These signals cause changes in the speeds of the steering and drive motor, producing one state that Grey

Walter called “steer hard, push gently,” and another called “steer gently, push hard.” The robot is still steering via Level 1, and driving via Level 0, so the overall result is the robot changing direction in various ways that eventually cause it to move away from the obstacle that depressed the shell.

```
//Level 3: The shell can temporarily override the light sensors.
int TimeConstant, Bumped; //Reaction time constant and threshold for contact.
task level_3(){
    while(true){
        until (Shell < Bumped);
        Vision = 0;
        while (Shell < Bumped){//Flicker between dark and bright.
            See = dark ;
            Wait(TimeConstant * Eye);//Itill flicker differently if it sees light.
            See = bright;
            Wait(TimeConstant * Eye * 2);
        }
        Vision = 1;
    }
}
```

7.12 THE MAIN TASK

7.12.1 Modular Design

In the code below a number of `# define` commands name the input and output ports. Code for Levels 0 through 3 – which are saved in separate files – is then included. The `main` task initializes motor speeds, sensor settings, timing constants, and the sensor values that define dark and bright. It also starts each of the tasks that define each level of our subsumption architecture. The code is organized level by level, to reflect the nature of our architecture. By deleting a Level’s `# include` command, and by deleting the code that initializes the level’s variables in the main task, one can study how the Tortoise behaves with a different version of its architecture (i.e., a version with one or more selected levels ablated).

```
//Tortoise NXT code
//Definitions in plain English
#define DriveMotor OUT_C
#define TurnMotor OUT_A
#define EyePort S1
```

```

#define Eye SENSOR_1
#define ShellPort S2
#define Shell SENSOR_2
#define PilotLight OUT_B
task main(){
    //Set up hardware.
    SetSensorType(EyePort, SENSOR_TYPE_LIGHT_INACTIVE);
    SetSensorMode(EyePort, SENSOR_MODE_RAW);
    SetSensorType(ShellPort, SENSOR_TYPE_LIGHT_ACTIVE);
    SetSensorMode(ShellPort, SENSOR_MODE_RAW);
    //Init level 0.
    DriveSpeed = 70;
    start level_0;
    //Init level 1.
    TurnSpeed = 40;
    start level_1;
    //Init level 2.
    Zero = 0;
    HalfDrive = 40; FullDrive = 60;
    HalfTurn = 7; FullTurn = 20;
    dark = 450; bright = 700;
    Vision = 1;
    start level_2;
    //Init level 3.
    TimeConstant = 5;
    Bumped = 620;
    start level_3;
}

```

7.13 OBSERVING TORTOISE BEHAVIOUR

7.13.1 Level 0

Let us first consider LEGO Tortoise performance as levels are added one by one. This is demonstrated in Video 7-1.mpg, available from the website that supports this book (<http://www.bcp.psych.ualberta.ca/~mike/BricksToBrains/>). The first behavioural segment in this video shows what occurs when only Level 0 is operating. The robot moves forward, in whatever direction the front wheel was pointing when the robot was activated. The robot is insensitive to the light in its environment.

7.13.2 Level 0 + Level 1

The next behavioural segment in the video illustrates the effect of adding Level 1 to Level 0. Level 1 rotates the front axle a full 360°. When combined with Level 0, this produces a wandering movement. Although the robot approaches the light in the video, this is merely accidental – the light sensor in the robot is not active at this time. As well, the robot bumps into the light, but its shell is also not active at this time. The robot, in fact, blindly wanders into the light, and then blindly wanders away from it.

7.13.3 Level 0 + Level 1 + Level 2

The next behavioural segment illustrates the robot’s behaviour when light sensitivity is added to the previous two levels of the subsumption architecture. Note, now, that the robot’s behaviour is much more “light directed.” First, rather than randomly wandering into the light, the robot moves directly toward it. Second, rather than randomly wandering away from the light, the robot circles the light at a respectful distance.

This version of the robot does not yet have obstacle avoidance activated. However, it appears that the robot is able to move away from the light when it is bumped. In actuality, this movement depends entirely on the front-wheel drive rotating away from the light when bright light is sensed. As well, the embodiment of the shell permits the robot to slide off an obstacle, much as the shape of a locomotive’s cowcatcher permits it to move obstacles out of a train’s path.

7.13.4 All Four Levels

The total LEGO Tortoise adds obstacle detection to the previous three levels. In essence, obstacle detection functions as follows: when the robot’s shell is depressed, light is momentarily ignored. The lower levels are manipulated by Level 3 to produce turning behaviour that moves the robot away from the obstacle. After a brief period of time, normal operations resume. The period of time during which the robot is insensitive to light depends upon the amount of light that is currently being sensed. In short, it “remembers” obstacles for a period of time, but then forgets them and returns to being a light-sensing robot.

The combination of the four levels can produce behaviour that appears to be much more complicated than one might predict from knowing about the construction and programming of the LEGO Tortoise. For instance, during the final behavioural segment of the video, a shoe is tossed at the robot, activating its shell. From an analytic perspective, the behaviour that ensues seems fairly elaborate. It is as if the robot

scurries away from the attack, hiding for a while under the tables that are also in the environment. When the coast appears clear, the robot is tempted out of its hiding spot by the light.

This raises one interesting theme that can be explored with the LEGO Tortoise: the differences between synthetic and analytic theories of its behaviour. From a synthetic approach, we have constructed the robot, and produced its program. The result is a fairly simple machine that uses a handful of loosely co-operating reflexes to navigate through its world. Any behaviour that we observe – simple or complex – is going to be explained by appealing to this knowledge. In contrast, the analytic approach only has available to it observations of robot behaviour, and must use these observations to infer internal processes. What kind of theory might this produce? Will it be more complicated than the synthetic one? How much will it have to change as more and more environments are explored? Let us examine some more robot behaviour to consider this issue in more detail.

7.14 THE TOTAL TORTOISE

7.14.1 Repeating History

In Chapter 6, we saw a number of photographs that are the records of the behaviour of Elsie and Elmer (Holland, 2003a). Can we reproduce functionally similar behaviour with the LEGO Tortoise?

7.14.2 Search for an Optimum

Video 7-2.mpg, available from the website that supports this book (<http://www.bcp.psych.ualberta.ca/~mike/BricksToBrains/>). demonstrates the behaviour of the LEGO Tortoise in a number of situations that were inspired by Grey Walter's studies (Grey Walter, 1963). The video begins with the *search for an optimum* (see Figure 6-2 and the discussion in Section 6.4). In this situation, the robot's environment consists of a single light bulb on the floor, illuminating an otherwise dark room. When placed in this situation, the behaviour of the original Tortoise was described as follows: "Attracted at first by a distant bright light, the creature ... circles round it at a respectful distance, exhibiting a search for optima rather than maxima – the idea of moderation of the classical philosophers."

The LEGO Tortoise generates similar behaviour. In the dark, it oscillates around, seeking light. When the periscope mirror points toward the light source, the robot quickly moves toward it. However, when it comes too close to the light, it is dazzled. It then proceeds to circle the light at a safe distance. Knowing how this fairly simple robot has been

constructed and programmed, how would you explain this behaviour? How do you think that this behaviour would be explained by someone who was not familiar with how the robot was constructed, and could only analyze what he or she observed?

7.14.3 Free Will

Now consider an environment in which two light sources are present on the floor. Grey Walter observed his Tortoise first circle one light, and then move to circle the other, demonstrating choice behaviour (see Figure 6-3 in Section 6.5), and rising above Buridan's ass. Similar choice behaviour is demonstrated by the LEGO Tortoise in the video's next segment. It begins by respectfully circling the light at the bottom of the video screen. When the circle is mostly complete, it quickly departs, and moves to the other light, which it also circles. This second "inspection" complete, it returns to the first light. "By scholastic definition the creature appears endowed with 'free will'. It approaches and investigates first one goal and then abandons this to investigate the other." Our synthetic methodology allows us to explain the robot behaviour without appealing to free will – but to what causes would an analytic approach appeal? Would they be the same as those appealed to in an analytic theory of Section 7.15.3?

7.14.4 Discernment

Grey Walter explored discernment by combining positive and negative tropisms. For example, he combined a single attracting light with a single repelling obstacle in one study (see Figure 6-4 in Section 6.6). His robot – and the LEGO Tortoise, as seen in the video – avoided the obstacle, maintaining a brief memory of its presence before proceeding to investigate the light. Again, consider the different accounts of this complex behaviour that would be produced by synthetic and analytic methodologies. Might analytic theories become more complex as behavioural complexity increases? Is this true of our synthetic theory?

7.14.5 Self-Recognition

Grey Walter's Tortoise's are perhaps most famous for performing the mirror dance (see Figure 6-6 in Section 6.8). When seeking light in front of a mirror, Elsie was attracted to the reflection of her pilot light. However, when the reflection was approached, her pilot light turned off (because the steering motor was off), causing new seeking behaviour to be produced. The result was a complex trajectory along a mirror that Grey Walter noted could be interpreted as evidence for self-awareness. In the

video, the LEGO Tortoise produces a similar mirror dance, in the dark and in the light. The dance stops as soon as the mirror is removed. We possess a simple, synthetic account of this behaviour. How complicated would an analytic theory of it be?

7.15 TORTOISE IMPLICATIONS

7.15.1 Grey Walter's Legacy

Grey Walter's Tortoises have been described as the first biologically inspired robots, and as the first example of behaviour-based or "new" robotics (Holland, 2003b).

Grey Walter's robots serve as inspirations for embodied cognitive science at a number of different levels. His general purpose was to create machines "that would imitate a living creature in performance, as distinguished from appearance" (Grey Walter, 1963, p. 122). Such imitation was to be produced by exploiting a small number of simple principles. "Two ideas, goal-seeking and scanning, had combined as the essential mechanical conception of a working model that would behave like a very simple animal" (p. 125). Furthermore, these simple principles were capable of producing emergent behaviour because of feedback between the robot and its environment. "This again illustrates an important general principle in the study of animal behaviour – that any psychological or ecological situation in which such a reflexive mechanism [feedback] exists, may result in behaviour which will seem, at least, to suggest self-consciousness or social consciousness" (p. 130).

In short, Grey Walter's work illustrated that seemingly complex behaviour, often attributed to complex internal processes, might actually be the result of simple internal processes interacting with a complex environment. This is one of the fundamental ideas of embodied cognitive science.

7.15.2 The LEGO Tortoise

The LEGO Tortoise is important because it permits us to obtain hands-on experience with Grey Walter's influential ideas.

First, when the robot is activated, we are armed with a great deal of knowledge about its structure and mechanisms, because we have created this machine. As a result, whenever we observe complex or surprising behaviour, we are in a position to explain it by appealing to these known mechanisms. Our synthetic approach should produce simpler accounts of this complex behaviour than would be achieved by analyzing the behaviour without having built the robot (Braitenberg, 1984).

Second, with the robot we can easily demonstrate environmental

contributions to behavioural complexity. That is, behavioural complexity should increase by leaving the machine alone, and by modifying its environment. This message was pioneered by early classical cognitive scientists (Simon, 1969), then largely ignored by classical cognitive science, but championed by embodied cognitive science (Clark, 1997, 2003; Dourish, 2001; Norman, 2002).

One example of exploring environmental complexity with the LEGO Tortoise was recently provided by one of my students. She created a square enclosure for the Tortoise, where each side was either opaque or mirrored. She manipulated environmental complexity by manipulating the number and location of mirrored sides. As more mirrors were added, the robot's environment became increasingly complicated because of the proliferation of reflections of its pilot lights. The result was a steady progression in the elaborateness of the mirror dance that the robot produced.

7.15.3 Degrees of Embodiment

The LEGO Tortoise also points us in the direction of the next issue to be explored in our study of embodied cognitive science using simple robots. It has been argued that robots can be differentiated in terms of their degrees of embodiment (Fong, Nourbakhsh, & Dautenhahn, 2003). A robot is not merely embodied by being constructed. Rather, its embodiment is reflected in the degree to which it can be perturbed by its environment, and can in turn affect its environment. Thus, the LEGO Tortoise is only moderately embodied: it can sense its environment, but only changes its environment accidentally (e.g., by bumping a light obstacle out of the way). In the next chapter, we will consider what might be gained by increasing embodiment, when we describe a robot that is no more complex than the Tortoise, but which is explicitly designed to modify its environment as it moves through it.