Introduction

Since the end of the Shakey project [Nilsson, 1984], an ever widening gap has opened up between theoretical work in Knowledge Representation and the practice of robotics. At the end of the Eighties, this trend culminated in the work of Brooks, who rejected the assumptions underlying the Shakey project altogether [Brooks, 1986], [Brooks, 1991].

> *The key problem I see with [all work in the style of Shakey] is that it relied on the assumption that a complete world model could be built internally and then manipulated.* [Brooks, 1991a, page 577]

As Brooks points out, a complete model of the world is hard for a robot to construct because,

> *The data delivered by sensors are not direct descriptions of the world as objects and their relationships [and] commands to actuators have very uncertain effects.* [Brooks, 1991b, page 5]

Without ignoring the lessons of the early Seventies, the nascent area of Cognitive Robotics [Lespérance, *et al.*, 1994] seeks to reinstate the ideals of the Shakey project, namely the construction of robots whose architecture is based on the idea of representing the world by sentences of formal logic and reasoning about it by manipulating those sentences. The chief benefits of this approach are,

- That it facilitates the endowment of a robot with the capacity to perform high-level reasoning tasks, such as planning, and
- that it makes it possible to formally account for the success (or otherwise) of a robot by appealing to the notions of correct reasoning and correct representation.

Where Brooks and his followers see residual adherence to the ideals of the Shakey project as a barrier to progress in AI, Cognitive Robotics laments the failure to pursue those ideals with sufficient zeal. Contrary to ill-informed opinion, logic is eminently suited to representing incomplete models of the world, which are the only models that can be built on the basis of the incomplete and noisy data delivered by a robot's sensors.

Unlike the Shakey project, contemporary Cognitive Robotics can draw on twenty five years of progress in the formalisation of common sense reasoning, specifically on representing and reasoning about action, change, space, and shape, and especially on reasoning with incomplete information. In addition, it benefits from advances in a number of enabling technologies, including logic programming techniques, algorithms from mainstream robotics, and low-cost mobile robotic platforms with substantial on-board computing power.

This paper concerns the representation of knowledge about the objects in a robot's environment, and how such knowledge is acquired. The main feature of this knowledge is its incompleteness and uncertainty, placing the robot in what McCarthy calls the *common sense informatic situation* [1989]. The treatment given in the paper is rigorously logical, but has been carried through to implementation on a real robot.

The paper is organised as follows. After outlining the basic idea of abductive sensor data assimilation, a general purpose formalism for representing action, continuous change, space and shape is introduced. This formalism is then used to represent the relationship between the actions of a particular (real) robot and the sensor data it receives from the world. The abductive task of assimilating sensor data is then straightforwardly characterised. The next topic to be considered is that of noise, which is taken to mean uncertainty in both sensor input and motor output. Two theorems are then developed which facilitate the design of map-building algorithms that can be executed on the robot. Finally, the results of preliminary experimentation with the robot are reported.

1 Assimilating Sensor Data

The key idea of this paper is to consider the process of assimilating a stream of sensor data as abduction. Given such a stream, the abductive task is to hypothesise the existence, shapes, and locations of objects which, given the output the robot has supplied to its motors, would explain that sensor data [Charniak & McDermott, 1985, page 455]. This is, in essence, the map building task for a mobile robot.

More precisely, if a stream of sensor data is represented as the conjunction Ψ of a set of observation sentences, the task is to find an explanation of Ψ in the form of a logical description (a map) Δ_M of the initial locations and shapes of a number of objects, such that,

$$\Sigma_B \wedge \Sigma_E \wedge \Delta_N \wedge \Delta_M \models \Psi$$

where,

- Σ_B is a background theory, comprising axioms for change (including continuous change), action, space, and shape,
- Σ_E is a theory relating the shapes and movements of objects (including the robot itself) to the robot's sensor data, and
- Δ_N is a logical description of the movements of objects, including the robot itself.

The exact form of these components is described in the next three sections, which present formalisms for representing and reasoning about action, change, space, and shape. In practice, as we'll see, these components will have to be split into parts for technical reasons.

Three major issues arise with this logical specification of the map building task: noisy data, incomplete information, and implementation.

- Σ_E does not have to assume a perfect correspondence between objects in the world and sensor data received from them, or a perfect correspondence between motor outputs and actual movements in the world. In practice, a noisy interface between world and robot must be assumed. Using the expressive power of first-order logic, the uncertainty resulting from such noise can be captured.
- Data in the common sense informatic situation is incomplete as well as noisy. In abductive terms, there will typically be many Δ_M's that could explain any given Ψ. For example, the robot may only receive sensor data from a small fraction of the total surface of an object, and be unable to tell whether the object is large or small. Again, using the expressive power of first-order logic, this incompleteness can be captured.
- This logical specification of the map building task must be rendered into an efficient implementation which can be executed by the onboard microprocessor of a mobile robot.

The provision of a logic-based theoretical account brings issues like noise and incompleteness into sharp focus, and permits their study within the same framework used to address wider epistemological questions in knowledge representation. It also enables the formal evaluation of algorithms for low-level motor-perception tasks by supplying a formalism in which these tasks can be precisely specified. These topics are discussed in detail in later sections.

2 Representing Action

The formalism used in this paper to represent action and change, including continuous change, is adapted from the circumscriptive event calculus presented in [Shanahan, 1995b], which in turn is based loosely on the formalism of Kowalski and Sergot [1986]. However, it employs a novel solution to the frame problem, inspired by the work of Kartha and Lifschitz [1995]. The result is a considerable simplification of the formalism in [Shanahan, 1995b].

Throughout the paper, the language of many-sorted first-order predicate calculus with equality will be used, augmented with circumscription [McCarthy, 1986], [Lifschitz, 1994]. Variables in formulae begin with lower-case letters and are universally quantified with maximum scope unless indicated otherwise.

In the event calculus, we have sorts for *fluents*, *actions* (or events), and *time points*. It's assumed that time points are interpreted by the reals, and that the usual comparative predicates, arithmetic functions, and trigonometric functions are suitably defined. The formula $HoldsAt(f,t)$ says that fluent f is true at time point t. The formulae $Initiates(a,f,t)$ and $Terminates(a,f,t)$ say respectively that action a makes fluent f true from time point t, and that a makes f false from t. The effects of actions are described by a collection of formulae involving *Initiates* and *Terminates*.

For example, if the term $Rotate(r)$ denotes a robot's action of rotating r degrees about some axis passing through its body, and the term $Facing(r)$ is a fluent representing that the robot is facing in a direction r degrees from North, then we might write the following *Initiates* and *Terminates* formulae.[1]

$$Initiates(Rotate(r1), Facing(r2), t) \leftarrow HoldsAt(Facing(r3), t) \wedge r2 = r3 + r1 \quad 2.1$$

$$Terminates(Rotate(r1), Facing(r2), t) \leftarrow HoldsAt(Facing(r2), t) \wedge r1 \neq 0 \quad 2.2$$

Once a fluent has been initiated or terminated by an action or event, it is subject to the common sense law of inertia, which is captured by the event calculus axioms to be presented shortly. This means that it retains its value (true or false) until another action or event occurs which affects that fluent.

A narrative of actions and events is described via the predicates *Happens*, $Initially_P$, and $Initially_N$. The formula $Happens(a,t)$ says that an action or event of type a occurred at time point t. Events are instantaneous. The formula $Initially_P(f)$ says that the fluent f is true from time point 0, and the formula $Initially_N(f)$ says that the fluent f is false from time point 0. Heres an example narrative.

[1]Rotation is treated as instantaneous here, and throughout the sequel. So long as the robot always finishes turning vefore it moves forwards, this idealization is acceptable.

$Initially_P(Facing(0))$ 2.3

$Happens(Rotate(90), 10)$ 2.4

$Happens(Rotate(-180), 20)$ 2.5

A theory will also include a pair of uniqueness-of-names axioms, one for actions and one fluents.[2]

$UNA(Facing)$ 2.6

$UNA(Rotate)$ 2.7

The relationship between *HoldsAt*, *Happens*, *Initiates*, and *Terminates* is constrained by the following axioms. Note that a fluent does not hold at the time of an action or event that initiates it, but does hold at the time of an action or event that terminates it.

$HoldsAt(f, t) \leftarrow Initially_P(f) \wedge \neg Clipped(0, f, t)$ EC1

$HoldsAt(f, t2) \leftarrow$ EC2
$\quad Happens(a, t1) \wedge Initiates(a, f, t1) \wedge$
$\quad tl < t2 \wedge \neg Clipped(t1, f, t2)$

$\neg HoldsAt(f, t) \leftarrow Initially_N(f) \wedge \neg Declipped(0, f, t)$ EC3

$\neg HoldsAt(f, t2) \leftarrow$ EC4
$\quad Happens(a, t1) \wedge Terminates(a, f, t1) \wedge$
$\quad t1 < t2 \wedge \neg Declipped(t1, f, t2)$

$Clipped(t1, f, t2) \leftrightarrow$ EC5
$\quad \exists a, t(Happens(a, t) \wedge$
$\quad (Terminates(a, f, t) \vee Releases(a, f, t)) \wedge$
$\quad t1 < t \wedge t < t2)$

$Declipped(t1, f, t2) \leftrightarrow$ EC6
$\quad \exists a, t(Happens(a, t) \wedge$
$\quad (Initiates(a, f, t) \vee Releases(a, f, t)) \wedge$
$\quad t1 < t \wedge t < t2)$

These axioms introduce a new predicate *Releases* [Kartha and Lifschitz, 1994]. The formula $Releases(a, f, t)$ says that action a exempts fluent f from the common sense law of inertia.[3] This non-inertial status is revoked as soon as the fluent is initiated or terminated once more. The use of this predicate will be illustrated shortly in the context of continuous change.[4]

Let the conjunction of (EC1) to (EC6) be denoted by EC. The circumscription policy to overcome the frame problem is the following. Given a conjunction N of *Happens*, $Initially_P$, and $Initially_N$ formulae, a conjunction E of *Initiates*, *Terminates* and *Releases* formulae, and a conjunction U of uniqueness-of-names axioms, we are interested in,

$CIRC[N\ ;\ Happens] \wedge CIRC[E\ ;\ Initiates, Terminates, Releases] \wedge$
$\quad U \wedge EC$

This formula embodies a form of the common sense law of inertia, and thereby solves the frame problem. To see why in detail, consult [Shanahan, 1997a]. The key to the solution is to put EC outside the scope of the circumscriptions, thus ensuring that the Hanks-McDermott problem is avoided [Hanks & McDermott, 1987].[5] In most cases, the two circumscriptions will yield predicate completions, making the overall formula manageable and intuitive.

---

[2] The UNA notation is taken from [Baker, 1989].

[3] The *Releases* predicate is related to Sandewall's idea of occlusion [Sandewall, 1994].

[4] Formula (2.2) above can be written using *Releases* instead of *Terminates*, so long as there is a domain constraint ensuring that the robot always has a unique orientation.

[5] This is related to Sandewall's idea of filtered preferential entailment [Sandewall, 1994].

For the example above, we have the following proposition. Let E be the conjunction of (2.1) with (2.2), let N be the conjunction of (2.3) to (2.5), and let U be the conjunction of (2.6) with (2.7).

Proposition 2.8.

$$
\begin{aligned}
&CIRC[N\ ;\ Happens] \wedge \\
&\ CIRC[E\ ;\ Initiates, Terminates, Releases] \wedge U \wedge EC \models \\
&\quad HoldsAt(Facing(r), t) \leftarrow \\
&\qquad (0 \leq t \leq 10 \wedge r = 0) \vee \\
&\qquad (10 < t \leq 20 \wedge r = 90) \vee \\
&\qquad (20 < t \wedge r = 270).
\end{aligned}
$$

Proof. See Appendix A.

3 Domain Constraints and Continuous Change

Two additional features of the calculus are important: the ability to represent domain constraints, and the ability to represent continuous change.

Domain constraints are straightforwardly dealt with in the proposed formalism. They are simply formulated as *HoldsAt* formulae with a single universally quantified time variable, and conjoined outside the scope of the circumscriptions along with EC. For example, the following domain constraint expresses the fact that the robot can only face in one direction at a time.

$$HoldsAt(Facing(r1), t) \wedge HoldsAt(Facing(r2), t) \rightarrow r1 = r2$$

In the event calculus, domain constraints are used to determine values for fluents that haven't been initiated or terminated by actions or events (non-inertial fluents) given the values of other fluents that have. (Domain constraints that attempt to constrain the relationship between inertial fluents can lead to inconsistency.)[6]

The issue of continuous change has been largely neglected in the design of formalisms for reasoning about action until recently [Sandewall, 1989], [Shanahan, 1990], [Miller, 1996], [Reiter, 1996]. In the present formalism, following [Shanahan, 1990], continuous change is represented through the introduction of a new predicate and the addition of an extra axiom. The formula $Trajectory(f1, t, f2, d)$ represents that, if the fluent $f1$ is initiated at time t, then after a period of time d the fluent $f2$ holds. We have the following axiom.

$$
\begin{aligned}
&HoldsAt(f2, t2) \leftarrow && \text{EC7} \\
&\quad Happens(a, t1) \wedge Initiates(a, f1, t1) \wedge \\
&\quad t1 < t2 \wedge t2 = t1 + d \wedge \\
&\quad Trajectory(f1, t1, f2, d) \wedge \neg Clipped(t1, f1, t2)
\end{aligned}
$$

Let CEC denote EC $\wedge$ (EC7), and U denote the conjunction of a set of uniqueness-of-names axioms. If R is the conjunction of a set of domain constraints and T is the conjunction of set of formulae constraining *Trajectory*, then we are interested in,

$$
\begin{aligned}
&CIRC[N\ ;\ Happens] \wedge CIRC[E\ ;\ Initiates, Terminates, Releases] \wedge \\
&\quad T \wedge R \wedge U \wedge CEC.
\end{aligned}
$$

For example, suppose the robot's repertoire of actions is expanded to include the actions *Go* and *Stop*. The *Go* action initiates a period of continuous change in the robot's location. The *Stop* action terminates such a period. For the present, the robot's location will be represented by the fluent $Location(Robot, p)$, where p is a pair of Cartesian co-ordinates the

[6]Note that $Initiates(a, F1, t) \rightarrow Initiates(a, F2, t)$ does *not* follow from $HoldsAt(F1, t) \rightarrow HoldsAt(F2, t)$.

form $\langle x, y\rangle$.[7] A constant velocity V is assumed in the following collection of formulae, which are intended to capture this example.

Let E be the conjunction of the following formulae.

$$\mathit{Initiates}(\mathit{Go}, \mathit{Moving}, t) \quad 3.1$$

$$\mathit{Releases}(\mathit{Go}, \mathit{Location}(\mathit{Robot}, p), t) \quad 3.2$$

$$\mathit{Terminates}(\mathit{Stop}, \mathit{Moving}, t) \quad 3.3$$

$$\mathit{Initiates}(\mathit{Stop}, \mathit{Location}(\mathit{Robot}, p), t) \leftarrow \mathit{HoldsAt}(\mathit{Location}(\mathit{Robot}, p), t) \quad 3.4$$

Let T be the following formula.

$$\begin{aligned}&\mathit{Trajectory}(\mathit{Moving}, t, \mathit{Location}(\mathit{Robot}, \langle x2, y2\rangle), d) \leftarrow \quad 3.5\\ &\quad \mathit{HoldsAt}(\mathit{Location}(\mathit{Robot}, \langle x1, y1\rangle), t) \wedge\\ &\quad \mathit{HoldsAt}(\mathit{Facing}(r), t1) \wedge\\ &\quad x2 = x1 + V.d.\mathit{Sin}(r) \wedge\\ &\quad y2 = y1 + V.d.\mathit{Cos}(r)\end{aligned}$$

Let R be the following domain constraint.

$$\mathit{HoldsAt}(\mathit{Location}(w, p1), t) \wedge \mathit{HoldsAt}(\mathit{Location}(w, p2), t) \rightarrow p1 = p2 \quad 3.6$$

Let U be the conjunction of the following uniqueness-of-names axioms.

$$\mathit{UNA}[\mathit{Location}, \mathit{Facing}, \mathit{Moving}] \quad 3.7$$

$$\mathit{UNA}[\mathit{Go}, \mathit{Stop}] \quad 3.8$$

Let N be the following narrative description.

$$\mathit{Initially}_P(\mathit{Location}(\mathit{Robot}, \langle 0, 0\rangle)) \quad 3.9$$

$$\mathit{Initially}_P(\mathit{Facing}(90)) \quad 3.10$$

$$\mathit{Happens}(\mathit{Go}, 10) \quad 3.11$$

$$\mathit{Happens}(\mathit{Stop}, 20) \quad 3.12$$

Now, given that the circumscriptions of E and N yield the predicate completions of *Happens*, *Initiates*, *Terminates*, and *Releases*, its a straightforward exercise to show that the recommended circumscription yields what we would expect.

Proposition 3.13.

$$\begin{aligned}&\mathit{CIRC}[N\ ;\ \mathit{Happens}] \wedge\\ &\mathit{CIRC}[E\ ;\ \mathit{Initiates}, \mathit{Terminates}, \mathit{Releases}] \wedge\\ &T \wedge R \wedge U \wedge \mathit{CEC} \models\\ &\quad \mathit{HoldsAt}(\mathit{Location}(\mathit{Robot}, \langle x, y\rangle), t) \leftrightarrow\\ &\qquad (0 \leq t \leq 10 \wedge x = 0 \wedge y = 0) \vee\\ &\qquad (10 < t \leq 20 \wedge x = V.(t - 10) \wedge y = 0) \vee\\ &\qquad (20 < t \wedge x = V.10 \wedge y = 0).\end{aligned}$$

Proof. See Appendix A.

Notice that we are at liberty to include formulae which describe triggered events in N. Here's an example of such a formula, which describes conditions under which the robot will collide with a wall lying on an East-West line 100 units north of the origin.

$$\begin{aligned}&\mathit{Happens}(\mathit{Bump}, t) \leftarrow\\ &\quad \mathit{HoldsAt}(\mathit{Moving}, t) \wedge\\ &\quad \mathit{HoldsAt}(\mathit{Facing}(r), t) \wedge\\ &\quad -90 < r < 90 \wedge\\ &\quad \mathit{HoldsAt}(\mathit{Location}(\mathit{Robot}, \langle x, 90\rangle), t)\end{aligned}$$

[7] Later, the *Location* fluent will be replaced by a fluent which captures spatial occupancy.

4 Representing Space and Shape

The formalism used in this paper to represent space and shape is adapted from [Shanahan, 1995a].[8] Space is considered a real-valued co-ordinate system.[9] For present purposes we can take space to be the plane $\Re \times \Re$, reflecting the fact that the robot under consideration only moves in two dimensions. A *region* is a subset of $\Re \times \Re$. A *point* is a member of $\Re \times \Re$. I will consider only interpretations in which points are interpreted as pairs of reals, in which regions are interpreted as sets of points, and in which the predicate $\in$ has its usual meaning.

Objects occupy open, path-connected regions. For example, the following formula describes an open circle of radius z units centred on the origin.

$$p \in Disc(z) \leftrightarrow Distance(p, \langle 0,0 \rangle) < z \qquad \text{Sp1}$$

Distance is a function yielding a positive real number, defined in the obvious way.

$$Distance(\langle x1,y1 \rangle, \langle x2,y2 \rangle) = \sqrt{(x1-x2)^2 + (y1-y2)^2} \qquad \text{Sp2}$$

The function *Bearing* is also useful.

$$\begin{aligned} & Bearing(\langle x1,y1 \rangle, \langle x2,y2 \rangle) = r \leftarrow \qquad \text{Sp3} \\ & \quad z = Distance(\langle x1,y1 \rangle, \langle x2,y2 \rangle) \land z \neq 0 \land \\ & \quad Sin(r) = \tfrac{x2-x1}{z} \land Cos(r) = \tfrac{y2-y1}{z} \end{aligned}$$

Using *Distance* and *Bearing* we can define a straight line as follows. The term $Line(p1,p2)$ denotes the straight line segment whose end points are $p1$ and $p2$. The *Line* function is useful in defining shapes with straight line boundaries.

$$\begin{aligned} & p \in Line(p1,p2) \leftrightarrow \qquad \text{Sp4} \\ & \quad Bearing(p1,p) = Bearing(p1,p2) \land \\ & \quad Distance(p1,p) \leq Distance(p1,p2) \end{aligned}$$

Spatial occupancy is represented by the fluent *Occupies.* The term $Occupies(w,g)$ denotes that object w occupies region g. No object can occupy two regions at the same time. This implies, for example, that if an object occupies a region g, it doesn't occupy any subset of g nor any superset of g. We have the following domain constraints.

$$\begin{aligned} & HoldsAt(Occupies(w,g1),t) \land \qquad \text{Sp5} \\ & \quad HoldsAt(Occupies(w,g2),t) \rightarrow g1 = g2 \end{aligned}$$

$$\begin{aligned} & HoldsAt(Occupies(w1,g1),t) \land \qquad \text{Sp6} \\ & \quad HoldsAt(Occupies(w2,g2),t) \land w1 \neq w2 \rightarrow \\ & \quad \neg \exists p [p \in g1 \land p \in g2] \end{aligned}$$

The first of these axioms captures the uniqueness of an objects region of occupancy, and the second insists that no two objects overlap.

The term $Displace(g, \langle x,y \rangle)$ denotes the result of displacing the region g by x units east and y units north.

$$\begin{aligned} & \langle x1,y1 \rangle \in Displace(g, \langle x2,y2 \rangle) \leftrightarrow \qquad \text{Sp7} \\ & \quad \langle x1-x2, y1-y2 \rangle \in g \end{aligned}$$

The *Displace* function is primarily used to describe motion: if an object moves, the region it occupies is displaced. But it can also be used to combine regions by taking their union (via disjunction). The following formula

---

[8] The *Location* fluent and *Shape* function from [Shanahan, 1995a] have been dispensed with.

[9] This decision does not in any way rule out the adoption of qualitative approaches to spatial reasoning, in the spirit of the Naive Physics Manifesto [Hayes, 1985]. The real-valued co-ordinate system can be thought of as underpinning these approaches.

defines a shape a little like the field of view through a pair of binoculars, formed from two overlapping circles.

$$p \in \mathit{TwoDiscs}(x) \leftrightarrow$$
$$p \in \mathit{Displace}(\mathit{Disc}(x), \langle -0.5x, 0\rangle) \vee$$
$$p \in \mathit{Displace}(\mathit{Disc}(x), \langle 0.5x, 0\rangle)$$

The final component of the framework is a means of default reasoning about spatial occupancy [Shanahan, 1995a]. Shortly, a theory of continuous motion will be described. This theory insists that, in order for an object to follow a trajectory in space, that trajectory must be clear. Accordingly, as well as capturing which regions of space are occupied, our theory of space and shape must capture which regions are unoccupied.

A suitable strategy is to make space empty by default. It's sufficient to apply this default just to the situation at time 0 – the common sense law of inertia will effectively carry it over to later times. The following axiom is required, which can be thought of as a *common sense law of spatial occupancy*.

$$\mathit{AbSpace}(w) \leftarrow \exists g(\neg \mathit{Initially}_N(\mathit{Occupies}(w,g))) \qquad \text{Sp8}$$

The predicate *AbSpace* needs to be minimised, with $\mathit{Initially}_P$ and $\mathit{Initially}_N$ allowed to vary. (Note that, from the event calculus axioms, for any w and g, $\mathit{Initially}_N(\mathit{Occupies}(w,g))$ implies $\neg\mathit{Initially}_P(\mathit{Occupies}(w,g))$.)

Where previously we were interested in $\mathit{CIRC}[N\ ;\ \mathit{Happens}]$, it's now convenient to split this circumscription into two. Given,

- the conjunction O of Axioms (Sp1) to (Sp8),
- a conjunction M of $\mathit{Initially}_P$ and $\mathit{Initially}_N$ formulae which mention only the fluent *Occupies*, and
- a conjunction N of *Happens* formulae and $\mathit{Initially}_P$ and $\mathit{Initially}_N$ formulae which don't mention the fluent *Occupies*, and
- conjunctions E, T, R, U, and *CEC* as described in the last section,

we are now interested in,

$$\mathit{CIRC}[O \wedge M\ ;\ \mathit{AbSpace}\ ;\ \mathit{Initially}_P, \mathit{Initially}_N] \wedge$$
$$\mathit{CIRC}[N\ ;\ \mathit{Happens}] \wedge$$
$$\mathit{CIRC}[E\ ;\ \mathit{Initiates}, \mathit{Terminates}, \mathit{Releases}] \wedge$$
$$T \wedge R \wedge U \wedge \mathit{CEC}.$$

5 Sensors and Motors: The Theory Σ_E

We now have the logical apparatus required to construct a formal theory of the relationship between a robot's motor activity, the world, and the robot's sensor data. For now we will assume perfect motors and perfect sensors. The issue of noise is dealt with in Section 7.

The robot used as an example throughout the rest of the paper is one of the simplest and cheapest commercially available mobile robotic platforms at the time of writing, namely the Rug Warrior described by Jones and Flynn [1993] (Figure 1). This is a small (approximately 20cm diameter), wheeled robot with a 68000 series microprocessor plus 32K RAM on board. It has a very simple collection of sensors. These include three bump switches arranged around its circumference, which will be our main concern here. In particular, we will confine our attention to the two forward bump switches, which, in combination, can deliver three possible values for the direction of a collision.

Needless to say, each different kind of sensor gives rise to its own particular set of problems when it comes to constructing Σ_E. The question of

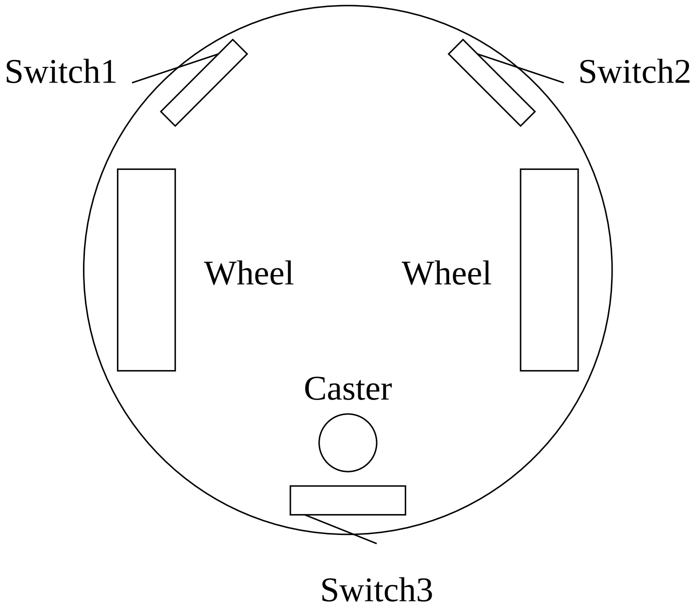

Figure 1: The Rug Warrior Robot from Above

noise is largely irrelevant when it comes to bump sensors. With infra-red proximity detectors, noise plays a small part. With sonar, the significance of noise is much greater. The use of cameras gives rise to a whole set of issues which are beyond the scope of this paper.

The central idea of this paper is the assimilation of sensor data through abduction. This is in accordance with the principle, "prediction is deduction but explanation is abduction" [Shanahan, 1989]. To begin with, we'll be looking at the predictive capabilities of the framework described. The conjunction of our general theory of action, change, space, and shape with the theory Σ_E, along with a description of the initial locations and shapes of objects in the world and a description of the robot's actions, should yield a description of the robot's expected sensory input. If prediction works properly using deduction in this way, the reverse operation of explaining a given stream of sensor data by hypothesising the locations and shapes of objects in the world is already defined. It is simply abduction using the same logical framework.

In the caricature of the task of assimilating sensor data presented in Section 1, the relationship between motor activity and sensor data was described by Σ_E. In practice, this theory is split into parts and distributed across different circumscriptions (see Section 3).

First, we have a collection of formulae which are outside the scope of any circumscription. Let B be the conjunction of CEC with Axioms (B1) to (B6) below. The robot is assumed to travel at a velocity of one unit of distance per unit of time.

$$UNA[Occupies, Facing, Moving, Blocked, Touching] \qquad \text{B1}$$

$$UNA[Rotate, Go, Stop, Bump, Switch1, Switch2] \qquad \text{B2}$$

$$\begin{aligned} &Trajectory(Moving, t, Occupies(Robot, g2), d) \leftarrow \qquad \text{B3} \\ &\quad HoldsAt(Occupies(Robot, g1), t) \wedge \\ &\quad HoldsAt(Facing(r), t) \wedge \\ &\quad g2 = Displace(g1, \langle d.Sin(r), d.Cos(r)\rangle) \end{aligned}$$

$$HoldsAt(Facing(r1), t) \land HoldsAt(Facing(r2), t) \rightarrow \quad \text{B4}$$
$$r1 = r2$$

$$HoldsAt(Blocked(w1, w2, r), t) \leftrightarrow \exists g1, g2[\quad \text{B5}$$
$$HoldsAt(Occupies(w1, g1), t) \land$$
$$HoldsAt(Occupies(w2, g2), t) \land$$
$$w1 \neq w2 \land$$
$$\exists z1[z1 > 0 \land \forall z2[z2 \leq z1 \rightarrow \exists p[p \in g2 \land$$
$$p \in Displace(g1, \langle z2.Sin(r), z2.Cos(r)\rangle)]]]]$$

$$HoldsAt(Touching(w1, w2, p), t) \leftrightarrow \exists g1, g2(\quad \text{B6}$$
$$HoldsAt(Occupies(w1, g1), t) \land$$
$$HoldsAt(Occupies(w2, g2), t) \land$$
$$w1 \neq w2 \land$$
$$\exists p1, p2(p \in Line(p1, p2) \land p \neq p1 \land p \neq p2 \land$$
$$\forall p3((p3 \in Line(p1, p) \land p3 \neq p) \rightarrow p3 \in g1) \land$$
$$\forall p3((p3 \in Line(p, p2) \land p3 \neq p) \rightarrow p3 \in g2)))$$

The fluent $Blocked(w1, w2, r)$ holds if object $w1$ cannot move any distance at all in direction r without overlapping with another object, namely $w2$. The fluent $Touching(w1, w2, p)$ holds if $w1$ and $w2$ are touching at point p. This is true if a straight line exists from $p1$ to $p2$ at a bearing r which includes a point $p3$ such that every point between $p1$ and $p3$ apart from $p3$ itself is in $g1$ and every point from $p2$ to $p3$ apart from $p3$ itself is in $g2$.

Next we have a collection of *Initiates*, *Terminates*, and *Releases* formulae. Let E be the conjunction of the following axioms (E1) to (E6). A *Bump* event occurs when the robot collides with something.

$$Initiates(Rotate(r1), Facing(r1 + r2), t) \leftarrow \quad \text{E1}$$
$$HoldsAt(Facing(r2), t)$$

$$Releases(Rotate(r1), Facing(r2), t) \leftarrow \quad \text{E2}$$
$$HoldsAt(Facing(r2), t) \land r1 \neq 0$$

$$Initiates(Go, Moving, t) \quad \text{E3}$$

$$Releases(Go, Occupies(Robot, g), t) \quad \text{E4}$$

$$Terminates(a, Moving, t) \leftarrow \quad \text{E5}$$
$$a = Stop \lor a = Bump \lor a = Rotate(r)$$

$$Initiates(a, Occupies(Robot, g), t) \leftarrow \quad \text{E6}$$
$$(a = Stop \lor a = Bump) \land$$
$$HoldsAt(Occupies(Robot, g), t)$$

Now we have a collection of formulae concerning the narrative of actions and events were interested in. This collection has two parts. Let N be N1 $\land$ N2. The first component part concerns triggered events. The events *Switch*1 and *Switch*2 occur when the robot's forward bump switches are tripped (see Figure 1). Let N1 be the conjunction of Axioms (H1) to (H3) below.[10]

$$Happens(Bump, t) \leftarrow \quad \text{H1}$$
$$(HoldsAt(Moving, t) \lor Happens(Go, t)) \land$$
$$HoldsAt(Facing(r), t) \land$$
$$HoldsAt(Blocked(Robot, w, r), t)$$

$$Happens(Switch1, t) \leftarrow \quad \text{H2}$$
$$Happens(Bump, t) \land$$
$$HoldsAt(Facing(r), t) \land$$
$$HoldsAt(Occupies(Robot, Displace(Disc(z), p1)), t) \land$$
$$HoldsAt(Touching(Robot, w, p2), t) \land$$
$$r - 90 \leq Bearing(p1, p2) < r + 12$$

[10] Both forward bump switches are tripped if the collision point is within approximatively 12° of the robot's bearing. This range was determined empirically.

$$Happens(Switch2, t) \leftarrow \quad \text{H3}$$
$$\quad Happens(Bump, t) \wedge$$
$$\quad HoldsAt(Facing(r), t) \wedge$$
$$\quad HoldsAt(Occupies(Robot, Displace(Disc(z), p1)), t) \wedge$$
$$\quad HoldsAt(Touching(Robot, w, p2), t) \wedge$$
$$\quad r - 12 \leq Bearing(p1, p2) < r + 90$$

The term $Occupies(Robot, Displace(Disc(z), p1))$ is employed in Axioms (H2) and (H3) to obtain the centre $p1$ of the region occupied by the robot, which can be thought of as its *location*. Note that Axiom (H1) caters for occasions on which the robot attempts to move when it is already blocked, as well as for occasions on which the robot's motion causes it to collide with something. In the former case, an immediate Bump event occurs, and the robot accordingly moves no distance at all.

For present purposes, the *Bump* event is somewhat redundant. In Axioms (E5) and (E6) it could be replaced by $Switch1$ and $Switch2$ events, and in Axioms (H2) and (H3) it could be simplified away. One reason not to abolish the *Bump* event is that, in principle, a collision could occur without the attendant sensor event if one of the bump switches were broken, say. Similarly, a sensor event could occur without a collision as its cause if a rain drop were to momentarily short a connection, for example.

Another reason is that abolishing the *Bump* event would violate a basic principle of the present approach, according to which the assumption of an external world governed by certain physical laws, a world to which its sensors have imperfect access, is built in to the robot. The robot's task is to do its best to explain its sensor data in terms of a model of the physics governing that world. In any such model, incoming sensor data is the end of the line, causally speaking. In the physical world, its not a sensor event that stops the robot but a collision with a solid object.

The second component of N is a description of the robot's actions. Suppose the robot behaves as illustrated in Figure 2. Let N2 be the conjunction of the following formulae, which represent the robot's actions up to the moment when it bumps into obstacle A.

$$Happens(Go, 0) \quad 5.1$$
$$Happens(Stop, 2.8) \quad 5.2$$
$$Happens(Rotate(90), 3.3) \quad 5.3$$
$$Happens(Go, 3.8) \quad 5.4$$

The final component of our theory is O $\wedge$ M, where M is a map of the robot's world and O is the conjunction of Axioms (Sp1) to (Sp8). Like N, M is conveniently divided into two parts. Let M be M1 $\wedge$ M2, where M1 is a description of the initial locations, shapes, and orientations (where applicable) of known objects, including the robot itself. For the example of Figure 2, M1 would be the conjunction of the following formulae.

$$Initially_P(Facing(80)) \quad 5.5$$
$$Initially_P(Occupies(Robot, Displace(Disc(0.5), \langle 1, 1 \rangle))) \quad 5.6$$

The form of M2 is the same as that of M1. However, when assimilating sensor data, M2 is supplied by abduction. For now though, lets look at the predictive capabilities of this framework, and supply M2 directly. Let M2 be the following formula, which describes the obstacle in Figure 2.

$$\exists g(Initially_P(Occupies(A, g)) \wedge \quad 5.7$$
$$\quad \forall x, y(\langle x, y \rangle \in g \leftrightarrow$$
$$\quad\quad 1 < x < 3 \wedge 3.5 < y < 4.5))$$

The following proposition says that, according to the formalisation, both bump switches are tripped at approximately time 5.5 (owing to a collision with obstacle A), and that the bump switches are not tripped at any other

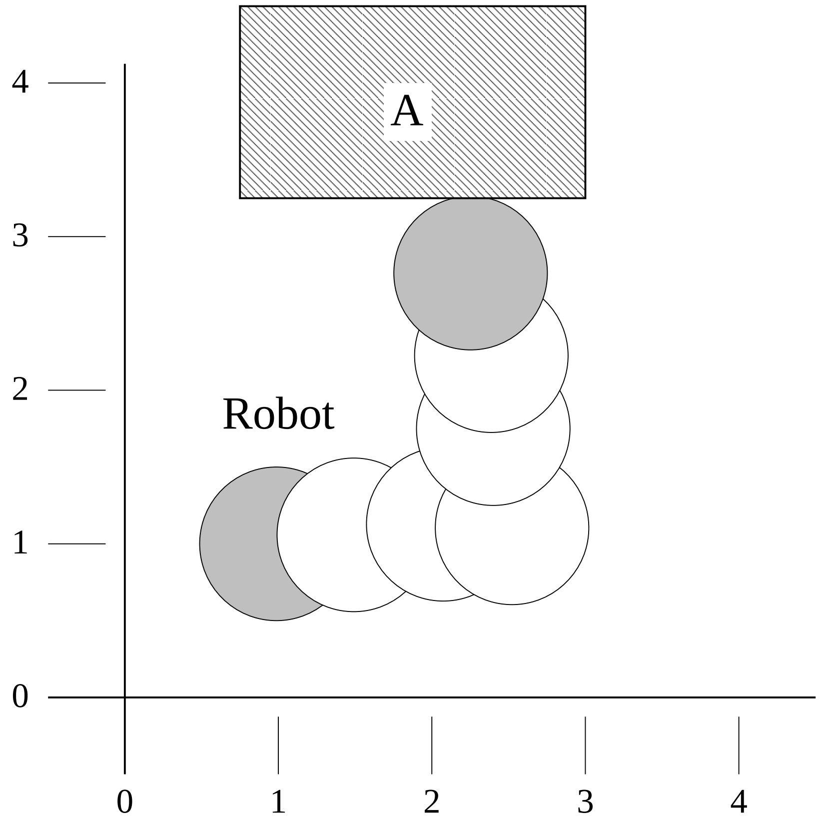

Figure 2: A Sequence of Robot Actions

time.

Proposition 5.8.

$$CIRC[O \wedge M1 \wedge M2 \ ; \ AbSpace \ ; \ Initially_P, Initially_N] \wedge$$
$$CIRC[N1 \wedge N2 \ ; \ Happens] \wedge$$
$$CIRC[E \ ; \ Initiates, Terminates, Releases] \wedge B \models$$
$$Happens(Switch1, T_{bump}) \wedge$$
$$Happens(Switch2, T_{bump}) \wedge$$
$$((Happens(Switch1, t) \vee Happens(Switch2, t)) \rightarrow$$
$$t = T_{bump})$$

where $T_{bump} = \frac{2.5+2.8\times Cos(80)}{Cos(-10)} + 3.8$

Proof. See Appendix A.

The process of assimilating sensor data is the reverse of that of predicting sensor data. As outlined in Section 1, the task is to postulate the existence, location, and shape of a collection of objects which would explain the robot's sensor data, given its motor activity.[11]

Let Ψ be the conjunction of a set of formulae of the form $Happens(Switch1, \tau)$ or $Happens(Switch2, \tau)$ where τ is a time point. What we want to explain is the *partial completion* of this formula, for reasons that will be made clear shortly. The only-if half of this completion is defined as follows.

Definition 5.9.

$$COMP[\Psi] \equiv_{def}$$
$$(Happens(a, t) \wedge (a = Switch1 \vee a = Switch2)) \rightarrow$$
$$\bigvee_{\langle\alpha,\tau\rangle\in\Gamma} a = \alpha \wedge t = \tau$$

where $\Gamma = \{\langle\alpha, \tau\rangle \mid Happens(\alpha, \tau) \in \Psi\}$.

Given Ψ, we're interested in finding conjunctions M2 of formulae in which each conjunct has the form,

$$\exists g(Initially_P(Occupies(\omega, g)) \wedge \forall p(p \in g \leftrightarrow \Pi))$$

where ω is an object constant and Π is any formula in which p is free, such that O $\wedge$ M1 $\wedge$ M2 is consistent and,

$$CIRC[O \wedge M1 \wedge M2 \ ; \ AbSpace \ ; \ Initially_P, Initially_N] \wedge$$
$$CIRC[N1 \wedge N2 \ ; \ Happens] \wedge$$
$$CIRC[E \ ; \ Initiates, Terminates, Releases] \wedge B \models$$
$$\Psi \wedge COMP[\Psi]$$

The partially completed form of the *Happens* formula on the right-hand-side of the turnstile eliminates anomalous explanations in which, for example, the robot encounters a phantom extra obstacle before the time of the first event in Ψ. If Ψ on its own were used instead of this partially completed formula, it would be possible to construct such explanations by shifting all the obstacles that appear in a proper explanation into new positions which take account of the premature interruption in the robot's path caused by the phantom obstacle.

Clearly, from Proposition 5.8, if Ψ is,

$$Happens(Switch1, T_{bump}) \wedge Happens(Switch2, T_{bump})$$

then (5.7) is an explanation that meets this specification.[12] Note that the symbol A in (5.7) (or rather its computational counterpart in the actual robot), when generated through the abductive assimilation of sensor

[11] In the present paper, it is assumed that all sensor data require explanation. However, to take account of glitches (as opposed to just noise), this requirement can be relaxed.

[12] It is assumed that the language includes an arbitrarily large set of unused constant symbols, from which ω is drawn. A similar assumption will apply throughout the sequel.

data, is *grounded* in Harnad's sense of the term [Harnad, 1990], at the same time as acquiring *meaning* through the theory. Furthermore, the theoretical framework within which such explanations are understood,

- Links the symbols that appear in them directly to a level of representation at which high-level reasoning tasks can be performed, and
- Licenses an account of the robot's success (or otherwise) at performing its tasks which appeals to the correctness of its representations and its reasoning processes.

However, (5.7) is just one among infinitely many possible explanations of this Ψ of the required form. An alternative explanation might involve the existence of an object of an entirely different shape. A bizarre example of an alternative explanation would be that the whole of space was occupied by a single object with a tunnel bored in it whose shape exactly matched that of the robot's path up to time T_{bump}.

In the specification of an abductive task like this, the set of explanations of the required form will be referred to as the *hypothesis space*. Its clear, in the present case, that some constraints must be imposed on the hypothesis space to eliminate bizarre explanations. Furthermore, even at a general mathematical level, the set of all explanations of the suggested form for a given stream of sensor data is hard to reason about, and computing a useful representation of such a set is infeasible. The general problem of the extravagant nature of the hypothesis space proposed above is tackled in the next section.

6 Boundaries

Many spatial representation techniques could be adapted for the purpose of rendering the hypothesis space more manageable (see [Davis, 1990, Chapter 6]). The one chosen here is based on the idea that a shapes boundary can be approximated to an arbitrary degree of precision by a series of straight lines.

The definition of a regions boundary is in terms of expansions of that region, defined as follows. The term *Expand*(g, d) denotes the region obtained by expanding g in all directions by a distance d.

$$p1 \in \mathit{Expand}(g,d) \leftrightarrow \exists p2[p2 \in g \wedge \mathit{Distance}(p1,p2) \leq d] \qquad \text{Bo1}$$

A set of points is a boundary of a region if it is outside that region, but any expansion of the region would engulf it. Note that this definition applies to any portion of the whole boundary, as well as to the whole boundary itself. The formula *Boundary*(g, l) denotes that l is a portion of the boundary of g.

$$\mathit{Boundary}(g,l) \leftrightarrow \forall p(p \in l \rightarrow (p \notin g \wedge \forall d(d > 0 \rightarrow p \in \mathit{Expand}(g,d)))) \qquad \text{Bo2}$$

The boundary of a shape will be described by a list of straight lines. The standard list functions *Nil* and *Cons* are introduced, and defined as follows.

$$\mathit{Member}(l1,c1) \leftrightarrow (c1 = \mathit{Cons}(l2,c2) \wedge (l1 = l2 \vee \mathit{Member}(l1,c2))) \qquad \text{Bo3}$$

$$\mathit{UNA}[\mathit{Cons}, \mathit{Nil}] \qquad \text{Bo4}$$

Every point on a boundary of the shape denoted by the term *Connect*(c) is also on one of the straight lines in the list c. So the term *Connect*(c)

denotes the largest polygon enclosed by the lines in c. If the lines in c don't form a closed curve then $Connect(c)$ is the empty set.

$$
\begin{array}{l}
p1 \in Connect(c) \leftrightarrow \\
\quad \exists g\ [p1 \in g \wedge \\
\quad\quad \forall p2, l\ [(Boundary(g, l) \wedge p2 \in l) \leftrightarrow \\
\quad\quad\quad \exists p3, p4\ [Member(Line(p3, p4), c) \wedge \\
\quad\quad\quad\quad p2 \in Line(p3, p4)]] \wedge \\
\quad\quad \exists p2, z\ \forall p3\ [p2 \in g \rightarrow p2 \in Displace(Disc(z), p3)]]
\end{array}
\qquad \text{Bo5}
$$

The last line of Axiom (Bo5) ensures that g is bounded, in the sense that a circle can be found which encloses it. This constrains g to be the inside of the shape described by c, and not the outside.

Given,

- the conjunction B of *CEC* with Axioms (B1) To (B6),
- the conjunction E of Axioms (E1) to (E6),
- the conjunction O of Axioms (Sp1) to (Sp8) with Axioms (Bo1) to (Bo5),
- a conjunction $M1$ of $Initially_P$ and $Initially_N$ formulae describing the initial locations, shapes, and orientations of known objects, including the robot itself,
- the conjunction $N1$ of Axioms (H1) to (H3),
- a conjunction $N2$ of *Happens* formulae describing the robot's actions, and
- a conjunction Ψ of formulae of the form $Happens(Switch1, \tau)$ or $Happens(Switch2, \tau)$,

we're now interested in finding conjunctions $M2$ of formulae in which each conjunct has the form,

$$Initially_P(Occupies(\omega, Connect(\lambda)))$$

where ω is an object constant, and λ is a term of the form,

$$Cons(Line(\rho1, \rho2), Cons(Line(\rho2, \rho3), \ldots Cons(Line(\rho_n, \rho_1), Nil)\ldots))$$

where $n \geq 3$ and $\rho_i \neq \rho_j$ for all $i \neq j$, such that $O \wedge M1 \wedge M2$ is consistent, and

$$
\begin{array}{l}
CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge \\
\ CIRC[N1 \wedge N2; Happens] \wedge \\
\ CIRC[E; Initiates, Terminates, Releases] \wedge B \models \\
\quad \Psi \wedge COMP[\Psi]
\end{array}
$$

Now, returning to the example of Figure 2, if we let $M1$ and $N2$ be the same formula as in the previous section, and let $M2$ be,

$$
\begin{array}{l}
Initially_P(Occupies(A, Connect(\\
\quad Cons(Line(\langle 1.0, 4.5\rangle, \langle 3.0, 4.5\rangle), \\
\quad\ Cons(Line(\langle 3.0, 4.5\rangle, \langle 3.0, 3.5\rangle), \\
\quad\quad Cons(Line(\langle 3.0, 3.5\rangle, \langle 1.0, 3.5\rangle), \\
\quad\quad\ Cons(Line(\langle 1.0, 3.5\rangle, \langle 1.0, 4.5\rangle)))))))))
\end{array}
\qquad 6.1
$$

then the following proposition is true.

Proposition 6.2.

$$
\begin{array}{l}
CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge \\
\ CIRC[N1 \wedge N2; Happens] \wedge \\
\ CIRC[E; Initiates, Terminates, Releases] \wedge B \models \\
\quad Happens(Switch1, T_{bump}) \wedge \\
\quad Happens(Switch2, T_{bump}) \wedge \\
\quad [[Happens(Switch1, t) \vee Happens(Switch2, t)] \rightarrow \\
\quad\quad t = T_{bump}]
\end{array}
$$

where $T_{bump} = \frac{2.5+2.8\times Cos(80)}{Cos(-10)} + 3.8$

Proof. See Appendix A.

From Proposition 6.2, we see that (6.1) is an explanation of the sensor data which meets the above specification.

We now have a specification of the abductive task of explaining sensor data which is susceptible to both implementation and formal analysis. In addition, this theoretical framework enables us to bring formal techniques to bear on issues such as the uncertainty resulting from noise, which is addressed in the next section.

7 Noise

The hallmark of the common sense informatic situation for a mobile robot is incomplete and uncertain knowledge of a world of spatio-temporally located objects. Incompleteness is a consequence of the robot's limited window on the world, and uncertainty results from noise in its sensors and actuators. This section deals with noise.

Both noisy sensors and noisy actuators can be captured using non-determinism. (An alternative is to use probability [Bacchus, *et al.*, 1995]). Here we'll only look at the uncertainty in the robot's location that results from its noisy motors. The robot's motors are "noisy" for various reasons. For example, the two wheels might rotate at slightly different speeds when the robot is trying to travel in a straight line, or the robot might be moving on a slope or a slippery surface. [13] Motor noise of this kind can be captured using a non-deterministic *Trajectory* formula, such as the following replacement for Axiom (B3).[14]

$$\begin{aligned} &\exists p[\,Trajectory(Moving, t,\\ &\qquad Occupies(Robot, Displace(g,p)), d) \wedge\\ &\quad Distance(p, \langle d.Sin(r), d.Cos(r)\rangle) \leq d.\varepsilon] \leftarrow\\ &\qquad HoldsAt(Occupies(Robot, g), t) \wedge\\ &\qquad HoldsAt(Facing(r), t) \end{aligned} \qquad \text{B7}$$

In effect, Axiom (B7) constrains the robot's location (the centre of the region it occupies) to be within an ever-expanding *circle of uncertainty* centred on the location it would be in if its motors weren't noisy.[15] The constant ε determines the rate at which this circle grows. Axiom (B8) below ensures that there are no discontinuities in the robot's trajectory. Without this axiom the robot would be able to leap over any obstacle which didn't completely cover the circle of uncertainty for its location. The term $Abs(d)$ denotes the absolute value of d.

$$\begin{aligned} &Trajectory(f, t, Occupies(x, Displace(g, p1)), d1) \rightarrow\\ &\quad \forall z\; [z > 0 \rightarrow \exists d\; \forall d2, p2\; [\\ &\qquad (d2 > 0 \wedge\\ &\qquad\; Abs(d2\text{-}d1) < d \wedge\\ &\qquad\; Trajectory(f, t, Occupies(x, Displace(g, p2)), d2)) \rightarrow\\ &\qquad\quad Distance(p1, p2) < z]] \end{aligned} \qquad \text{B8}$$

[13] The Rug Warrior's wheels are fitted with shaft encoders, which can be used to reduce the non-determinism in the robot's movements. These are ignored in the present analysis.

[14] The *Rotate* action could also be made non-deterministic, yielding only a range of possible values for the robot's subsequent orientation.

[15] Note that, while objects occupy open subsets of $\Re^2$, regions of uncertainty are closed.

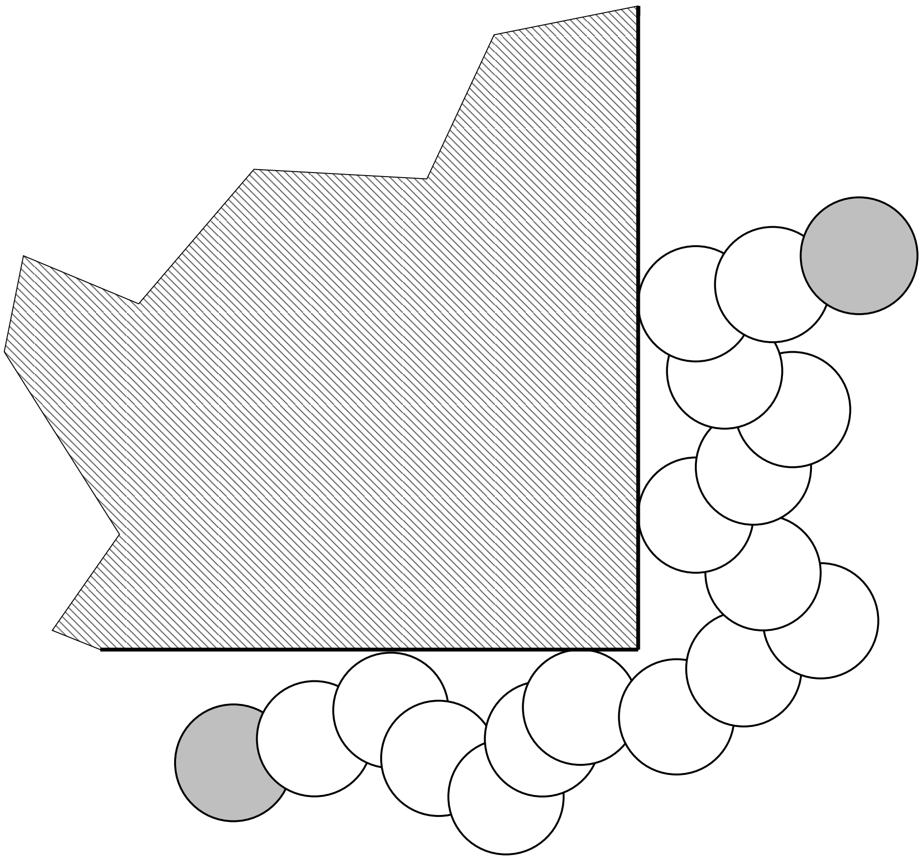

Figure 3: The Robot Explores a Corner

Figure 3 shows the robot exploring the corner of an obstacle. Figure 4 shows the evolution of the corresponding circle of uncertainty, highlighting the points where the robot changes direction.

Figure 4 is somewhat misleading, however. Consider Figure 5. On the left, the evolution of the circle of uncertainty for the robot's location is shown. In the middle, three potential locations are shown for the three changes of direction. Although these locations all fall within the relevant circles of uncertainty, the robot could never get to the third location from the second. This is because, as depicted on the right of the figure, in any given model the circle of uncertainty for the robot's location at the end of a period of continuous motion can only be defined relative to its actual location at the start of that period. This can be verified by inspecting Axioms (B7) and (B8).

The relative nature of the evolution of the circle of uncertainty means that the robot can acquire a detailed knowledge of some area $A1$ of its environment, then move to another area $A2$ which is some distance from $A1$, and acquire an equally detailed knowledge of $A2$. The accumulated uncertainty entails only that the robot is uncertain of where $A1$ is relative to $A2$. This natural feature of the formalisation conforms with what we would intuitively expect given the robot's informatic situation.

Non-determinism is a potential source of difficulty for the abductive approach to explanation. Even with a precise and complete description of the initial state of the world, including all its objects and their shapes, a

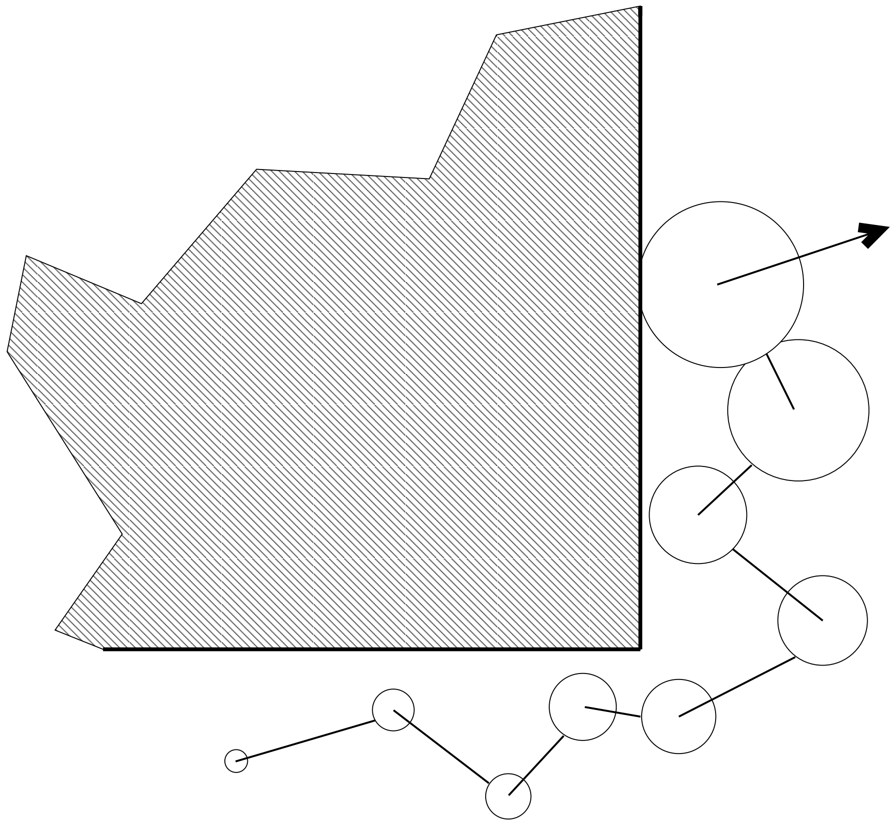

Figure 4: The Evolution of the Circle of Uncertainty

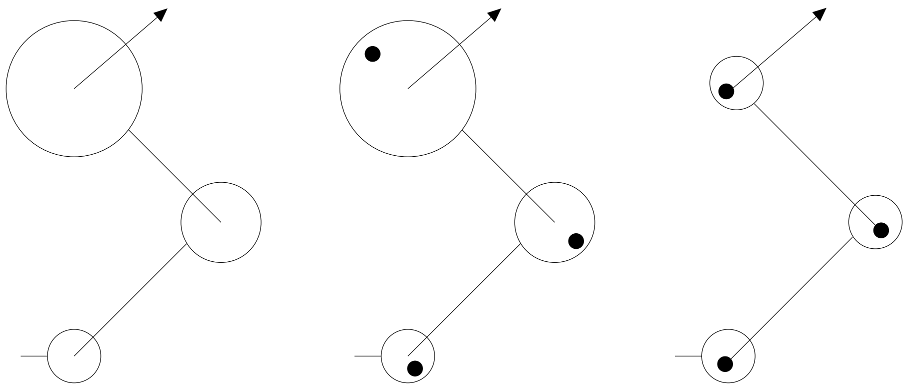

Figure 5: The Circle of Uncertainty is Relative Not Absolute

non-deterministic theory incorporating a formula like (B7) will not yield the exact times at which collision events occur. Yet the sensor data to be assimilated has precise times attached to it. Fortunately we can recast the task of assimilating sensor data as a form of *weak abduction* so that it yields the required results. Intuitively what we want to capture is the fact that without the hypothesised objects, the sensor data could not have been received. This is analogous to the consistency-based approach to diagnosis proposed by Reiter [1987].

Definition 7.1.
Given,

- the conjunction B of *CEC* with Axioms (B1), (B2), and (B4) to (B8),
- the conjunction E of Axioms (E1) to (E6),
- the conjunction O of Axioms (Sp1) to (Sp8) with Axioms (Bo1) to (Bo5),
- a conjunction $M1$ of $Initially_P$ and $Initially_N$ formulae describing the initial locations, shapes, and orientations of known objects, including the robot itself,
- the conjunction $N1$ of Axioms (H1) to (H3),
- a conjunction $N2$ of *Happens* formulae describing the robot's actions, and
- a conjunction Ψ of formulae of the form $Happens(Switch1, \tau)$ or $Happens(Switch2, \tau)$

an *explanation* of Ψ is a conjunction $M2$ of formulae in which each conjunct has the form,

$$Initially_P(Occupies(\omega, Connect(\lambda)))$$

where ω is an object constant, and λ is a term of the form,

$$Cons(Line(\rho1, \rho2), Cons(Line(\rho2, \rho3), \ldots Cons(Line(\rho_n, \rho_1), Nil)\ldots))$$

where $n \geq 3$ and $\rho_i \neq \rho_j$ for all $i \neq j$, such that $O \wedge M1 \wedge M2$ is consistent, and,

$$\begin{aligned}&CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge \\ &\quad CIRC[N1 \wedge N2; Happens] \wedge \\ &\quad CIRC[E; Initiates, Terminates, Releases] \wedge B \not\models \\ &\qquad \neg[\Psi \wedge COMP[\Psi]].\end{aligned}$$

To illustrate the new definition, suppose the robot behaves as illustrated in Figure 6. Let $N2$ be the conjunction of the following formulae, which represent the robot's actions up to and including the time it bumps into obstacle A.

$$Happens(Go, 0) \tag{7.2}$$

$$Happens(Stop, 2.1) \tag{7.3}$$

Let $M1$ be the conjunction of the following formulae.

$$Initially_P(Facing(0)) \tag{7.4}$$

$$Initially_P(Occupies(Robot, Displace(Disc(0.5), \langle 2, 1\rangle))) \tag{7.5}$$

Let $M2$ be the following formula.

$$\begin{aligned}&Initially_P(Occupies(A, Connect(\\ &\quad Cons(Line(\langle 1.0, 4.5, \langle 3.0, 4.5\rangle), \\ &\quad\; Cons(Line(\langle 3.0, 4.5\rangle, \langle 3.0, 3.5\rangle), \\ &\quad\;\; Cons(Line(\langle 3.0, 3.5\rangle, \langle 1.0, 3.5\rangle), \\ &\quad\;\;\; Cons(Line(\langle 1.0, 3.5\rangle, \langle 1.0, 4.5\rangle)))))))))\end{aligned} \tag{7.6}$$

In the noise-free case, the robot would collide with A at time 2.0. However, let's assume the collision takes place at time 2.1. Let Ψ be the conjunction of the following formulae.

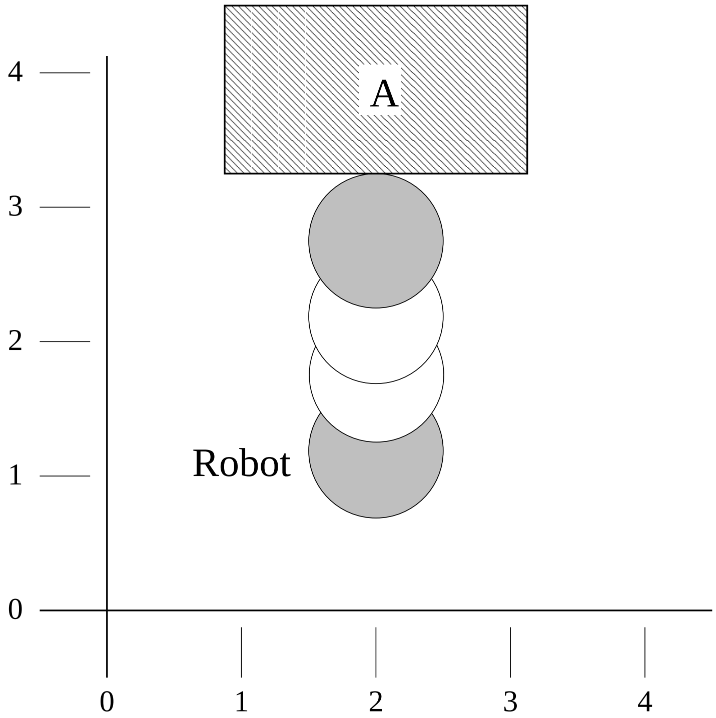

Figure 6: The Robot Collides with an Obstacle

$Happens(Switch1, 2.1)$ 7.7

$Happens(Switch2, 2.1)$ 7.8

Let ε be 0.25. The following proposition says that $M2$ is indeed an explanation of Ψ according to the new definition.

Proposition 7.9.

$$CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge$$
$$CIRC[N1 \wedge N2; Happens] \wedge$$
$$CIRC[E; Initiates, Terminates, Releases] \wedge B \not\models$$
$$\neg[\Psi \wedge COMP[\Psi]].$$

Proof. See Appendix A.

There will, naturally, be many explanations for any given Ψ which meet Definition 7.1. For the example above, an explanation which postulated an object of the same shape but 0.1 units of distance further north would have sufficed. A standard way to treat multiple explanations in abductive knowledge assimilation is to adopt their disjunction [Shanahan, 1997a, Chapter 17]. This has the effect of smothering any explanations which are stronger than necessary, such as those which postulate superfluous obstacles. The disjunction of all explanations of Ψ is the *cautious explanation* of Ψ .

A variety of *preference relations* over explanations can also be introduced, which make the abductive process more selective. For example, it might be reasonable to assume that nearby collision points indicate the presence of a single object. This is captured in the following definitions.

Definition 7.10. Two objects are *separated by x* units if there is a line of length x which has points in common with the boundaries of both objects, and there is no shorter such line.

Definition 7.11. An explanation M_1 is *preferable* to an explanation M_2 if the number of objects in M_1 which are separated from another object by less than MIN units is less than the number of such objects in M_2.

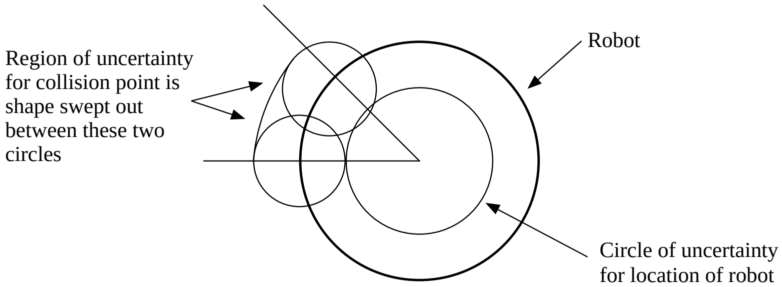

Figure 7: The Region of Uncertainty for a Collision Point

A *preferred* explanation is one for which no preferable explanation exists. Such preference relations are a topic for further study, and will not feature in the rest of the paper.

The following theorem establishes that the above definition of an explanation is equivalent to the deterministic specification offered in the last section when ε is 0. Let B_{det} be the conjunction of *CEC* with Axioms (B1) to (B6).

Definition 7.12. A formula M is a *complete spatial description* if the region occupied by each object mentioned in M is the same in every model of,

$$CIRC[O \wedge M; AbSpace; Initially_P, Initially_N].$$

Theorem 7.13. If $\varepsilon = 0$ and $M1$ is a complete spatial description, then $M2$ is an explanation of Ψ if and only if $O \wedge M1 \wedge M2$ is consistent and,

$$\begin{aligned}&CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge\\&CIRC[N1 \wedge N2; Happens] \wedge\\&CIRC[E; Initiates, Terminates, Releases] \wedge B_{det} \models\\&\quad \Psi \wedge COMP[\Psi].\end{aligned}$$

Proof. See Appendix B.

Note that, as illustrated in Figure 7, while the location of the robot at any time falls within a circle of uncertainty, the corresponding region of uncertainty for a collision point is kidney-shaped, since the collision point could be anywhere along a portion of the circumference of the robot. (Recall that the forward bump switches deliver three possible values for the direction of a collision.)

In the next section, two results are established which support the claim that the abductive specification of the task of sensor data assimilation offered here gives the results we would expect, and which also make the task of building algorithms based on this specification straightforward.

8 Two Theorems

The two theorems in this section are an attempt to characterise the abductive explanations defined in the previous section in terms which appeal more directly to the information available to any map-building algorithm which might be executed on board the robot. This is done by picking out the regions of uncertainty within which the collision points corresponding to a stream of switch events must lie.

Let B be the conjunction of *CEC* with Axioms (B1), (B2), and (B4) to (B8). Let E be the conjunction of axioms (E1) to (E6). Let O be the conjunction of Axioms (Sp1) to (Sp8) with Axioms (Bo1) to (Bo5). Let $M1$

be a conjunction of $Initially_P$ and $Initially_N$ formulae describing the initial locations, shapes, and orientations of known objects, including the robot itself. Let $N1$ be the conjunction of Axioms (H1) to (H3).

Let $N2$ be a conjunction of *Happens* formulae describing the robot's actions which is equivalent to the formula,

$\phi_1 \wedge \phi_2 \wedge \ldots \wedge \phi_n$

where each ϕ_i has the form $Happens(\alpha_i, \tau_i)$, and,

- for all $i < n$, $\tau_i \leq \tau_{i+1}$ if α_i is Go and α_{i+1} is *Stop*,
- for all $i < n$, $\tau_i < \tau_{i+1}$ if α_i is Go and α_{i+1} is not *Stop*,
- α_1 is Go , and
- for all $i < n$, α_{i+1} is $\begin{cases} Stop & if\ \alpha_i\ is\ Go \\ Rotate(\theta) & if\ \alpha_i\ is\ Stop \\ Go & if\ \alpha_i\ is\ Rotate(\theta) \end{cases}$

$N2$ describes a finite sequence of actions of the form,

Go, *Stop*, *Rotate*, *Go*, *Stop*, *Rotate*, *Go*,...

The i^{th} action in $N2$ occurs before the $i+1^{th}$ action, unless the i^{th} action is a *Go* and the $i+1^{th}$ is a *Stop*, in which case they can occur simultaneously. (A *Go* and a *Stop* will occur simultaneously if the robot tries to move when it is already pressed up against an obstacle.) Let Ψ be a conjunction of the form,

$\psi_1 \wedge \psi_2 \wedge \ldots \wedge \psi_m$

where each ψ_i is either a *type 1 datum* of the form,

$Happens(Switch1, \tau_i)$

or a *type 2 datum* of the form,

$Happens(Switch2, \tau_i)$

or a *type 3 datum* of the form,

$Happens(Switch1, \tau_i) \wedge Happens(Switch2, \tau_i)$

such that for all $i < m$, $\tau_i < \tau_{i+1}$.

Definition 8.1. $N2$ *tracks* Ψ if for every occurrence in Ψ of a formula of the form,

$Happens(\alpha, \tau)$

where a is either *Switch*1 or *Switch*2, there is a corresponding occurrence in $N2$ of the formula,

$Happens(Stop, \tau)$.

We will assume that $N2$ tracks Ψ.[16] This is the case if the robot's control mechanism ensures that it stops moving immediately when it bumps into something. This makes the theorems below much easier to state and prove, since it enables us to ignore the possibility of the robot's wheels spinning while it is pressed up against an obstacle.

Let $T_{go}(i)$ and $T_{stop}(i)$ be respectively the times of the i^{th} *Go* action in $N2$ and the i^{th} *Stop* action in $N2$. Where no such action exists, these functions are undefined.

The *MT* function, defined below, yields the amount of time the robot has been in motion between the j^{th} and the k^{th} *Stop* actions. The radius of the circle of uncertainty for the robot's location at the time of the k^{th} *Stop* action relative to its location at the time of the j^{th} *Stop* action is proportional to this value.

[16] In practice, of course, there's always a small delay between the *Bump* event and the *Stop* event. This delay is so small that it can safely be ignored.

Definition 8.2. The function MT (for Motion Time) is defined as follows.

$MT(j,k) = \sum_{j<i\leq k} T_{stop}(i) - T_{go}(i)$

Note that $MT(0,k)$ is the amount of time the robot has been in motion up to the time of the k^{th} *Stop* action.

The RO function, defined below, yields the orientation of the robot at a given time.

Definition 8.3. The function RO (for Robot Orientation) is defined as follows.

$RO(\tau) = \theta_0 + \sum_{i\leq R} \theta_i$

where R is the number of *Rotate* actions in $N2$ before τ, the i^{th} *Rotate* action in $N2$ is *Happens*(*Rotate*(θ_i), τ_i), and θ_0 is the robot's initial orientation according to $M1$.

The RL function, defined below, yields the centre of the circle of uncertainty for the robot's location at the time of the k^{th} *Stop* action given that its location is r at the time of the j^{th} *Stop* action.

Definition 8.4. The function RL (for Robot Location) is defined as follows.

$RL(\rho,j,k) = \rho + \sum_{j<i\leq k} \rho_i$

where,

- ρ_i is $\langle \delta_i.Sin(\theta_i),\ \delta_i.Cos(\theta_i)\rangle$,
- δ_i is $T_{stop}(i) - T_{go}(i)$, and
- θ_i is $RO(T_{go}(i))$.

Note that, if ρ is the robot's initial location, $RL(\rho,0,k)$ is the centre of the circle of uncertainty for the robot's location at the time of the k^{th} *Stop* action.

The BD function, defined below, yields the direction of the centre of the region of uncertainty for a collision (see Figure 6).

Definition 8.5. The functions BD_{min} and BD_{max} (for minimum and maximum Bump Direction) are defined as follows.

$$BD_{min}(n,\theta) = \begin{cases} \theta - 90 & \textit{if } n = 1 \\ \theta + 12 & \textit{if } n = 2 \\ \theta - 12 & \textit{if } n = 3 \end{cases}$$

$$BD_{max}(n,\theta) = \begin{cases} \theta - 12 & \textit{if } n = 1 \\ \theta + 90 & \textit{if } n = 2 \\ \theta + 12 & \textit{if } n = 3 \end{cases}$$

Let η_i be the type of datum ψ_i and let τ_i be the time of that datum. A bump trail, defined formally below, is a sequence of points, one for each collision featured in Ψ. The i^{th} point in this sequence ρ_i is a possible location, according to $N2$ and the axioms constraining the robot's motion, for the robot at the time of the i^{th} collision, given that the robot was at *ind*(ρ, i-1) at the time of the i-1^{th} collision (see Figure 6).

Definition 8.6. A sequence of points ρ_1 to ρ_m is a *bump trail* if,

- the distance from ρ_1 to the point $RL(\rho_0,0,k)$ is less than or equal to $\varepsilon.MT(0,k)$, where $T_{stop}(k)$ is τ_1 and ρ_0 is the robot's initial location according to $M1$, and
- for any i, $0 < i < m$, the distance from ρ_{i+1} to the point $RL(\rho_i,j,k)$ is less than or equal to $\varepsilon.MT(j,k)$, where $T_{stop}(j)$ is τ_i and $T_{stop}(k)$ is τ_{i+1}.

Now we define the relation OTR, which is used to state two theorems which characterise the set of explanations of a given stream of sensor data directly in terms of the robot's actions, and which justify the map building algorithm presented in the next section.

Definition 8.7. The relation OTR (for Obstacle Touches Robot) is defined as follows.

$$
\begin{aligned}
&OTR(p1, r1, r2) \equiv_{def} \\
&\quad \exists p2[r1 \leq Bearing(p1, p2) < r2 \wedge \\
&\qquad Distance(p1, p2) = 0.5 \wedge \\
&\qquad \exists w, g[Initially_P(Occupies(w, g)) \wedge \\
&\qquad\quad w \neq Robot \wedge \\
&\qquad\quad p2 \in Boundary(g) \wedge \\
&\qquad\quad \neg\exists p[p \in Line(p1, p2) \wedge p \in g]]]
\end{aligned}
$$

Less formally, $OTR(p1, r1, r2)$ is true if there is a line from $p1$ to $p2$ such that,

- $p2$ is at a bearing of between $r1$ and $r2$ degrees from $p1$ and 0.5 units away from $p1$,
- $p2$ is on the boundary of an object, and
- no point on the line from $p1$ to $p2$ is inside that object.

$OTR(p, r1, r2)$ is true if there is an object with a boundary which would touch the robot's circumference at a suitable point if the robot were located at p. The last of the above conditions ensures that the robot touches the outside of the object rather than the inside. The first theorem using the OTR relation sets out a necessary condition for$M2$ to be an explanation.

Theorem 8.8. If $M2$ is an explanation of Ψ then there exists a bump trail ρ_1 to ρ_m such that, for any i, $0 < i \leq m$,

$$
\begin{aligned}
&O \wedge M1 \wedge M2 \models \\
&\quad OTR(\rho_i, BD_{min}(\eta_i, RO(\tau_i)), BD_{max}(\eta_i, RO(\tau_i))).
\end{aligned}
$$

Proof. See Appendix B.

The second theorem partially defines the sufficient conditions for $M2$ to be an explanation. Only a partial characterisation is possible using OTR alone. A complete characterisation would have to take account of the constraints imposed on the spatial occupancy of hypothesised objects by the path along which the robot has moved without bumping into anything.

Theorem 8.9. $M2$ is an explanation of Ψ if $O \wedge M1 \wedge M2$ is consistent, there exists a bump trail ρ_1 to ρ_m such that, for any i, $0 < i \leq m$,

$$
\begin{aligned}
&O \wedge M1 \wedge M2 \models \\
&\quad OTR(\rho_i, BD_{min}(\eta_i, RO(\tau_i)), BD_{max}(\eta_i, RO(\tau_i)))
\end{aligned}
$$

and,

$$
\begin{aligned}
&CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge \\
&\quad CIRC[N1 \wedge N2; Happens] \wedge \\
&\quad CIRC[E; Initiates, Terminates, Releases] \wedge B \not\models \\
&\qquad \neg COMP[\Psi].
\end{aligned}
$$

Proof. See Appendix B.

A similar proof to that of Theorem 8.9 establishes the following consistency result.

Theorem 8.10. If $O \wedge M1 \wedge M2$ is consistent then so is,

$$
\begin{aligned}
&CIRC[O \wedge M1 \wedge M2; AbSpace; Initially_P, Initially_N] \wedge \\
&\quad CIRC[N1 \wedge N2; Happens] \wedge \\
&\quad CIRC[E; Initiates, Terminates, Releases] \wedge B.
\end{aligned}
$$

Proof. See Appendix B.

9 An Algorithm for Sensor Data Assimilation

One approach to building an implementation based on the the logical framework presented in this paper would be to isolate a fragment of the theory which could be executed as a logic program. A body of techniques for abductive logic programming has been developed, and these could be used to implement the abductive approach to sensor data assimilation [Kakas, Kowalski & Toni, 1993]. Taking a logic programming approach would make clear the sense in which the logical account given above supplies the meaning of the representations employed by the robot.

An alternative approach is to tailor-make algorithms for specific tasks, such as sensor data assimilation, whose correctness with respect to the logical account can be demonstrated. This is the methodology I will adopt here, and the logic programming approach is left for further research.

In both approaches, the products of the abductive process should be considered as general purpose knowledge with a declarative meaning. Theorem proving techniques, such as those employed in logic programming, are still applicable to the subsequent processing of this knowledge, in order to construct plans or to answer queries, for example. A tailor-made algorithm for sensor data assimilation, in so far as it can be validated in terms of the theoretical framework supplied, can be thought of as an efficient but highly specialised theorem prover.

The two theorems in the previous section can be used to justify a variety of algorithms for constructing a map of the robot's environment given a stream of sensor data. The algorithm described in this section works with a grid of squares superimposed on $\Re^+ \times \Re^+$. [17] Each square is denoted by a pair of co-ordinates, each of which is a natural number. This grid is represented in a two-dimensional *occupancy array* M (see [Davis, 1990, Section 6.2.1]). If $M[x,y] = c$, where c is either Black, White or Grey, then the square $\langle x, y \rangle$ is coloured c. Here is the main algorithm.

1. **Procedure** *ColourMap*
2. Let every element of M be *grey*
3. $P[0]$:= the robot's initial location according to $M1$
4. $j := 0$
5. **For** $i = 1$ to m
6. Find k such that the k^{th} *Stop* action in $N2$ occurs at τ_i
7. $PlotPath(P[i-1], q, j, k)$
8. $p := RL(q, k-1, k)$
9. $e := \varepsilon.MT(k-1, k)$
10. $b1 := BDmin(\eta_i, RO(\tau_i))$; $b2 := BDmax(\eta_i, RO(\tau_i))$
11. Choose $P[i]$, x, and y such that $PossSquare(P[i], p, b1, b2, e, x, y)$
12. $ColourPath(q, P[i])$
13. **If** $M(x,y) = white$
14. **Then Exit**
15. **Else** $M(x,y) := black$

[17] Without loss of generality, we can work in one quadrant of the plane.

16. $j := k$
17. **End** *For*
18. **Output** M

This non-deterministic procedure colours every square either *black*, *white* or *grey*, where *black* represents that the corresponding square region of $\Re^+ \times \Re^+$ is occupied by some object and *white* represents that it is empty. A boundary between two black squares is considered occupied while a boundary between a black and a white square is considered unoccupied, although in the algorithm description, a square region is assumed to include its boundaries (so a boundary is included in two squares and a corner is included in four). Let μ be the the length of the sides of a square.

Definition 9.1. The point $\langle a, b \rangle$ is *included* in the square $\langle x, y \rangle$ if and only if $x.\mu \leq a \leq (x+1).\mu$ and $y.\mu \leq b \leq (y+1).\mu$.

Let Ψ, $N2$, and $M1$ be defined as in the previous section, along with τ_i, η_i, and the functions *RL*, *MT*, BD_{min} and BD_{max}.

Some explanation of the algorithm is in order. To begin with all squares are coloured grey. The outer loop works through each of the m collisions in Ψ in the order in which they occurred, assigning a location to the robot for each one. This sequence of locations corresponds to a bump trail, as defined in the previous section, and is recorded in the array P. For each collision, a suitable path is first plotted which takes the robot up to the site of its last turn before that collision (Line 7).

The procedure *PlotPath* is defined as follows. This procedure non-determin-istically plots a path from point $p1$, which is robot's location at the time of the j^{th} *Stop* action, to $p2$, which is its location at the time of the k-1^{th} *Stop* action. The intermediate *Stop* actions are those which don't correspond to collisions.

1. **Procedure** $PlotPath(p1, p2, j, k)$
2. **For** $l = j + 1$ to $k - 1$
3. $p := RL(p1, l\text{-}1, l)$
4. $e := \varepsilon.MT(l\text{-}1, l)$
5. Choose any $p2$ whose distance from p is less than or equal to $\varepsilon.MT(j, k)$
6. $ColourPath(p1, p2)$
7. $p1 := p2$
8. **End** For

The path computed by *PlotPath* cuts through a region of space which is big enough for the robot to pass through. This region has to be coloured white, so that no attempt is made to place object inside it. This is the purpose of the procedure *ColourPath*, which is defined as follows.

1. **Procedure** $ColourPath(p1, p2)$
2. **For** each x and y such that $Covered(x, y, p1, p2)$
3. **If** $M[x, y] = black$
4. **Then Exit**
5. **Else** $M[x, y] := white$
6. **End** For

$Covered(x, y, p1, p2)$ is true if and only if the square $\langle x, y\rangle$ includes a point less than ζ units of distance from any point on the straight line from $p1$ to $p2$, where ζ is the radius of the robot.

Returning to the main procedure, the robot has now been taken up to the site of its last change of direction before a collision. Next, a point is chosen for its location at the time of the collision such that the robot will abut a square (Line 11). This square can then be coloured black. The boolean function $PossSquare(p1, p2, b1, b2, e, x, y)$, which picks out candidate squares, returns *True* if,

- the distance from $p1$ to $p2$ is less than or equal to e ,
- a straight line emanating from $p1$ at a bearing b, where $b1 \leq b < b2$ crosses a boundary and enters the square $\langle x, y\rangle$ at a distance of 0.5 units, and
- $M[x, y] \neq white$

and returns *False* otherwise.

If *ColourMap* terminates with output M, any squares in M that are still grey can be coloured either black or white, possibly to conform with some preference relation over explanations such as the example in Section 7. If *ColourMap* exits via Line 14 or via Line 4 of *ColourPath*, then its non-deterministic choices have not led to a consistent colouring of M. By augmenting the algorithm with a suitable search strategy, alternative choices can be explored.[18]

The following theorems relate this algorithm to the abductive characterisation of sensor data assimilation offered in the preceding sections. First we need to be able to convert the output of *ColourMap* to a logical form.

Definition 9.2. If M is an occupancy array, let $Exp^{\star}(M)$ denote the set of all conjunctions of formulae in which each conjunct has the form,

$Initially_P(Occupies(\omega, Connect(\lambda)))$

where ω is an object constant, and λ is a term of the form,

$Cons(Line(\rho1, \rho2), Cons(Line(\rho2, \rho3), ... Cons(Line(\rho_n, \rho_1), Nil)...))$

where $n \geq 3$ and $\rho_i \neq \rho_j$ for all $i \neq j$, such that the term $Line(\rho_i, \rho_{i+1})$ occurs in $Exp^{\star}(M)$ if and only if the line from ρ_i and ρ_{i+1} is a boundary shared by one black square and one grey or white square according to M.

Clearly the members of $Exp^{\star}(M)$ are all logically equivalent, modulo changes in the names of objects. Accordingly, let's suppose the existence of some standard ordering over the members of any given $Exp^{\star}(M)$, and let $Exp(M)$ denote the first in this ordering.

As usual, let B be the conjunction of *CEC* with Axioms (B1), (B2), and (B4) to (B8). Let E be the conjunction of axioms (E1) to (E6). Let O be the conjunction of Axioms (Sp1) to (Sp8) with Axioms (Bo1) to (Bo5). And let $N1$ be the conjunction of Axioms (H1) to (H3).

The following theorem expresses the soundness of the algorithm.

Theorem 9.3. If *ColourMap* terminates with output M then $Exp(M)$ is an explanation of Ψ.

Proof. See Appendix B.

The *ColourPath* procedure above assumes that the robot always travels in a straight line (although the bearing of that line and how far the

[18] Although each non-deterministic choice of location in the algorithm has uncountably many possibilities, there are only finitely many squares that each location can fall within. So an algorithm that explores the whole search space in finite time may be possible.

robot travels in a given time is uncertain). The algorithm misses explanations which depend crucially on the possibility, allowed by Axioms (B7) and (B8), of the robot curving around an obstacle to avoid it. The following formula, which is a strengthening of Axiom (B8), reflects this limitation, and permits us to state a theorem expressing the extent to which the algorithm is complete.

$$\exists p2, p3\ \forall d\ [\mathit{Trajectory}(f, t, \mathit{Occupies}(x, \mathit{Displace}(g, p1)), d) \rightarrow p1 \in \mathit{Line}(p2, p3)] \qquad \text{B9}$$

Definition 9.4. An explanation $M2$ of Ψ is *curve-free* if,

$$\mathit{CIRC}[O \wedge M1 \wedge M2; \mathit{AbSpace}; \mathit{Initially}_P, \mathit{Initially}_N] \wedge \mathit{CIRC}[N1 \wedge N2; \mathit{Happens}] \wedge \mathit{CIRC}[E; \mathit{Initiates}, \mathit{Terminates}, \mathit{Releases}] \wedge B \wedge (B9) \nvDash \neg \mathit{COMP}[\Psi]$$

Theorem 9.5. For every occupancy array M such that $\mathit{Exp}(M)$ is a curve-free explanation of Ψ, M is output by some execution of *ColourMap*.

Proof. See Appendix B.

The final question, to be briefly considered here, is which explanations are excluded by the demands on the form of an explanation made by Theorem 9.5. Clearly, for a given square-size μ , explanations for Ψ can exist which are not of the form $\mathit{Exp}(M)$ for any M. Any explanation which postulated the existence of a triangular object would be an example. On the other hand, it's also clear that there's a sense in which, by making μ small enough, any object's shape can be approximated to any desired degree of accuracy.

But the value of μ is not the only consideration here. The radius of the robot and the value of ε, the uncertainty factor in the robot's location, are also crucial. To see this, suppose ε is zero, and let the radius of the robot be 0.1μ Now suppose the robot collides with an obstacle, rotates $180°$ moves forward 0.5μ units and collides with another obstacle. Clearly no explanation of the required form exists for these collisions, because no suitable pair of squares can exist with boundaries sufficiently close together. Given the radius of the robot, even the uncertainty in the whereabouts on the robot's circumference of the collision point doesn't permit a suitable square to be found. An appropriate choice of μ for a given ε should render such cases pathological, but the mathematical basis for this claim remains to be worked out.

10 Experiments with the Robot

This section reports the results of some preliminary experimentation with the Rug Warrior robot based on the algorithm described in the preceding section. The implementation is rather crude, and should be regarded as nothing more than a demonstration of the feasibility of robotics research in the advocated style.

Control of the robot was achieved with a simple two-layered Brooks-style behaviour-based architecture [Brooks, 1986]. A *Wander* behaviour and an *Avoid* behaviour were implemented. The *Wander* behaviour executes a random walk around the environment. When this behaviour is active, the robot moves forwards for a certain distance, subject to random variation, then rotates a certain amount, also subject to random variation, then moves forwards again, and so on.

The *Avoid* behaviour is triggered when one or more of the forward bump

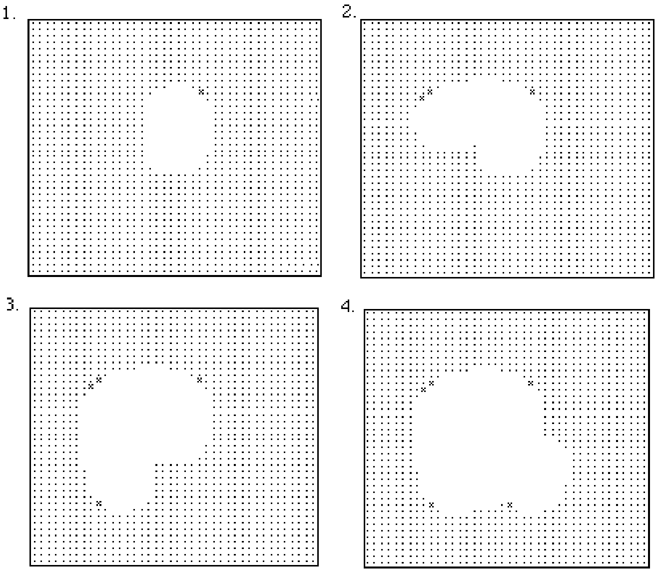

Figure 8: The Evolution of the Occupancy Array

switches is tripped. When the *Avoid* behaviour is active, after backing a small distance away from the (presumed) obstacle, the robot rotates a certain amount, subject again to random variation. Control is then relinquished, and the *Wander* behaviour becomes active once more.

In the background, a process is run continuously which records the robot's actions and the sensor events that occur. This data is subsequently processed by a program which incorporates the sensor data assimilation algorithm proposed in the previous section, and which generates an occupancy array. In the present implementation, this processing is done off-board, although this purely for convenience in program development and debugging. Both components – the behaviour-based control module and the sensor data assimilation module – are written in C, the former in a dialect which includes multi-tasking features which facilitate the construction of behaviour-based control systems. The control component comprises about 300 lines of code, and the assimilation program about 900 lines.

The present implementation of the sensor data assimilation algorithm performs a negligible amount of search, and is accordingly not always successful at explaining a given set of sensor data. It always chooses the centre of the circle of uncertainty for the robot's location when finding a path from one collision site to the next. Only when choosing the robot's location at the time of a collision does the algorithm exploit the existence of the circle of uncertainty by nudging the robot's location so that it abuts a square. The point on the robot's circumference which touches this square depends on whether the collision in question involved just Switch 1, just Switch 2, or both switches. In the first two cases, the chosen point is respectively $-45°$ and $45°$ from the robot's bearing. In the third case, the chosen point is directly ahead of the robot in the direction in which it is facing.

A small amount of search is carried out to determine the choice of abutting squares. If, because of occupancy data already present in the array, a location for the robot cannot be found which is within the square in which the centre of the circle of uncertainty for its location lies, then the surrounding four squares – north, east, south, and west – are also tried.

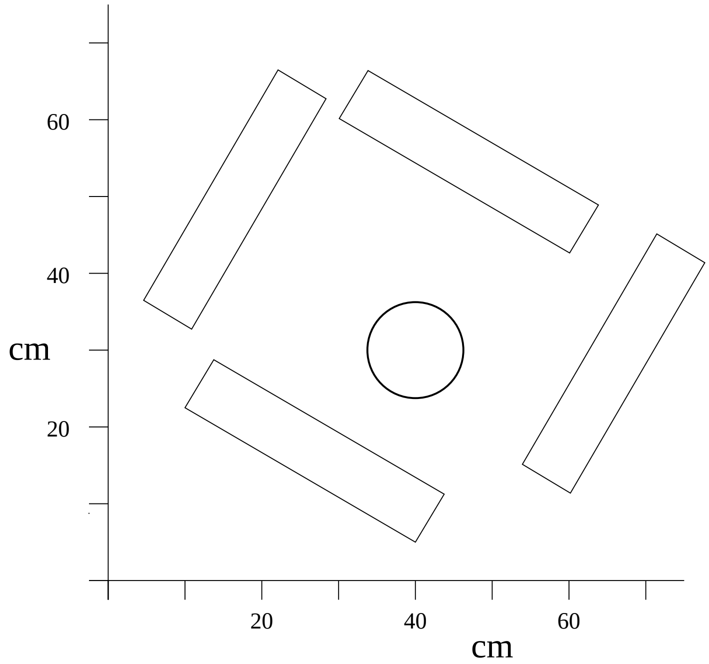

Figure 9: An Aerial View of the Robot

Figure 8 shows the evolution of the occupancy array generated by a sample run of the robot as it was moving about inside a pen made up from four box files. The array is 40 squares by 40 squares, and each square is 0.25 units of distance on each side. (Recall that one robot radius is the unit of distance.) The value of ε was 1.0. The dots denote grey squares (neither occupied nor unoccupied), the crosses denote black squares, and the blank area comprises white squares. The robot's actual environment is depicted in Figure 9, which is a scale drawing traced from an aerial photograph. As Figure 8 illustrates dramatically, with only a few collisions to go on, the robot will acquire more knowledge about empty space than about occupied space. However, given the severe limitations of the robot's perceptual apparatus, this is unsurprising.

The sample run shown is rather unimpressive since, although thirty actions were performed, only five *Bump* events took place. By allowing the implementation to perform more search, it is anticipated that much longer runs will be feasible, and more detailed maps will be produced. But the issue of future improvements in the implementation, though important, is a distraction from the real point here, which is that the occupancy array depicted has a logical interpretation according to which it is an explanation, in the formal sense defined in the foregoing sections, of the robot's sensor data given its actions. The benefits of this are twofold. First, the relationship between the robot's model of the world and the world itself is precisely defined. Second, a wealth of techniques and results from the field of Knowledge Representation is made available for the robot's further development.

Concluding Remarks

In the paper accompanying his 1991 Computers and Thought Award Lecture, Brooks remarked that,

> *[The field of Knowledge Representation] concentrates much of its energies on anomalies within formal systems which are never used for any practical task.* [Brooks, 1991a, page 578]

This paper should be construed as an answer to Brooks. According to the logical account given in this paper, a robot's incoming sensor data is filtered through an abductive process based on a framework of innate concepts, namely space, time, and causality.[19] The development of a rigorous, formal account of this process bridges the gap between theoretical work in Knowledge Representation and practical work in robotics, and opens up a great many possibilities for further research. The following issues, in ascending order of importance, are particularly pressing.

- The development of a more qualitative approach to spatial representation, possibly using a framework like that described by Randell *et al.* [1992], but augmented with the ability to handle certain kinds of metric information.
- The assimilation of sensor data from moving objects, such as humans, animals, or other robots. Movable obstacles should also be on the agenda.
- The assimilation of richer sensor data than that supplied by the Rug Warriors simple bump switches.
- The control of the robot via the model of the world it acquires through abduction.

The last of these issues, namely robot control, deserves some comment. Although theres nothing particularly remarkable about the map-building capabilities of the Rug Warrior implementation described in Section 10, which serves mainly to illustrate the theoretical and methodological ideas whose promotion is the paper's main purpose, its design does suggest a sharply bipartite architecture which differs markedly from those hinted at in most current work in the Cognitive Robotics vein [Lespérance, *et al.*, 1994], [Kowalski, 1995], [Poole, 1995].

The architecture of the Rug Warrior implementation is behaviour-based in the purest sense, and yet serves the purpose of building a symbolic model of the world for which a rigorous denotational account can be supplied. In a sense, this architecture accommodates two extremes in robot design – the behaviour-based approach and the Cognitive Robotics approach – but makes no concessions to do so. At present, the world model constructed by the robot is a passive by-product of its activity. But the purity of this bipartite architecture can be maintained when the model plays a more active role.

This is accomplished by using the model to inform the decisions of a rational core which attempts to orchestrate the talents the robot has at its

[19] This is somewhat reminiscent of Kant, according to whom, "the natural world as we know it, the whole content of our experience, is thoroguhly conditioned by [certain] features; our experience is essentially experience of spatio-temporal world of law-governed objects conceived as distinct from our temporally successive experiences of them" [Strawson, 1966, page 21].

disposal, in the form of a repertoire of Brooks-style behaviours, in order to achieve its goals. The behaviour-based component of the architecture needn't be under the direct control of the rational component, but can remain fully autonomous. The aim of the rational, goal-achieving component of the architecture is only to perturb these behaviours in ways conducive to the achievement of its goals. Although the very idea of a goal to some extent compromises the behaviour-based approach to robotics, this proposal preserves one of its essential insights, which is that tight coupling between sensors and effectors is the way to achieve robustness in control.

Acknowledgements

The inspiration for Cognitive Robotics comes from Ray Reiter and his colleagues at the University of Toronto. Neelakantan Kartha patiently explained his formalisation of actions to me at IJCAI 95 over many cups of tea, and this led to the frame problem solution presented in Section 2. Thanks to Rob Miller for careful proof reading and valuable feedback. Thanks to Peter Cheung of the Department of Electrical Engineering at Imperial College for financial support for the purchase of the robot. Finally, thanks to Chitta Baral and Paulo Santos for their questions and suggestions during the ETAI discussion period. Most of the work reported in this paper was undertaken while the author was an EPSRC Advanced Research Fellow.

References

[Bacchus, *et al.*, 1995] F.Bacchus, J.Y.Halpern, and H.J.Levesque, Reasoning about Noisy Sensors in the Situation Calculus, *Proceedings IJCAI 95*, pp. 1933-1940.

[Baker, 1989], A.B.Baker, A Simple Solution to the Yale Shooting Problem, *Proceedings 1989 Knowledge Representation Conference*, pp. 11-20.

[Brooks, 1986] R.A.Brooks, A Robust Layered Control System for a Mobile Robot, *IEEE Journal of Robotics and Automation*, vol 2, no 1 (1986), pp. 14-23.

[Brooks, 1991a] R.A.Brooks, Intelligence Without Reason, *Proceedings IJCAI 91*, pp. 569-595.

[Brooks, 1991b] R.A.Brooks, Artificial Life and Real Robots, *Proceedings of the First European Conference on Artificial Life (1991)*, pp 3-10.

[Charniak & McDermott, 1985] E.Charniak and D.McDermott, *Introduction to Artificial Intelligence*, Addison-Wesley (1985).

[Davis, 1990] E.Davis, *Representations of Commonsense Knowledge*, Morgan Kaufmann (1990).

[Hanks & McDermott, 1987] S.Hanks and D.McDermott, Nonmonotonic Logic and Temporal Projection, *Artificial Intelligence*, vol 33 (1987), pp. 379-412.

[Harnad, 1990] S.Harnad, The Symbol Grounding Problem, *Physica D*, vol 42 (1990), pp. 335-346.

[Hayes, 1985] P.J.Hayes, The Second Naive Physics Manifesto, in *Formal Theories of the Commonsense World*, ed. Hobbs J.R. and Moore R.C., Ablex (1985), pp. 1-36.

[Jones & Flynn, 1993] J.L.Jones and A.M.Flynn, *Mobile Robots: Inspiration to Implementation*, A.K.Peters (1993).

[Kakas, Kowalski & Toni, 1993] A.C.Kakas, R.A.Kowalski and F.Toni, Abductive Logic Programming, *Journal of Logic and Computation*, vol 2, no 6 (1993), pp. 719-770.

[Kartha & Lifschitz, 1994] G.N.Kartha and V.Lifschitz, Actions with Indirect Effects (Preliminary Report), *Proceedings 1994 Knowledge Representation Conference (KR 94)*, pp. 341-350.

[Kartha & Lifschitz, 1995] G.N.Kartha and V.Lifschitz, A Simple Formalization of Actions Using Circumscription, *Proceedings IJCAI 95*, pp. 1970-1975.

[Kowalski, 1995] R.A.Kowalski, Using Meta-Logic to Reconcile Reactive with Rational Agents, in *Meta-Logics and Logic Programming*, ed. K.R.Apt and F.Turini, MIT Press (1995), pp. 227-242.

[Kowalski & Sergot, 1986] R.A.Kowalski and M.J.Sergot, A Logic-Based Calculus of Events, *New Generation Computing*, vol 4 (1986), pp. 67-95.

[Lespérance, *et al.*, 1994] Y.Lespérance, H.J.Levesque, F.Lin, D.Marcu, R.Reiter, and R.B.Scherl, A Logical Approach to High-Level Robot Programming: A Progress Report, in *Control of the Physical World by Intelligent Systems: Papers from the 1994 AAAI Fall Symposium*, ed. B.Kuipers, New Orleans (1994), pp. 79-85.

[Lifschitz, 1994] V.Lifschitz, Circumscription, in *The Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3: Nonmonotonic Reasoning and Uncertain Reasoning*, ed. D.M.Gabbay, C.J.Hogger and J.A.Robinson, Oxford University Press (1994), pp. 297-352.

[McCarthy, 1986] J.McCarthy, Applications of Circumscription to Formalizing Common Sense Knowledge, *Artificial Intelligence*, vol 26 (1986), pp. 89-116.

[McCarthy, 1989] J.McCarthy, Artificial Intelligence, Logic and Formalizing Common Sense, in *Philosophical Logic and Artificial Intelligence*, ed. R.Thomason, Kluwer Academic (1989), pp. 161-190.

[Miller, 1996] R.S.Miller, A Case Study in Reasoning about Actions and Continuous Change, *Proceedings ECAI 96*, pp. 624-628.

[Nilsson, 1984] N.J.Nilsson, ed., *Shakey the Robot*, SRI Technical Note no. 323 (1984), SRI, Menlo Park, California.

[Poole, 1995] D.Poole, Logic Programming for Robot Control, *Proceedings IJCAI 95*, pp. 150-157.

[Randell, *et al.*, 1992] D.A.Randell, Z.Cui and A.G.Cohn, A Spatial Logic Based on Regions and Connection, *Proceedings 1992 Knowledge Representation Conference (KR 92)*, pp. 165-176.

[Reiter, 1987] R.Reiter, A Theory of Diagnosis from First Principles, *Artificial Intelligence*, vol 32 (1987), pp. 57-95.

[Reiter, 1996] R.Reiter, Natural Actions, Concurrency and Continuous Time in the Situation Calculus, *Proceedings 1998 Knowledge Representation Conference (KR 98)*, pp. 176-185.

[Sandewall, 1989] E.Sandewall, Combining Logic and Differential Equations for Describing Real-World Systems, *Proceedings 1989 Knowledge Representation Conference (KR 89)*, pp. 412-420.

[Sandewall, 1994] E.Sandewall, *Features and Fluents: The Representation of Knowledge about Dynamical Systems, Volume 1*, Oxford University Press, 1994.

[Shanahan, 1989] M.P.Shanahan, Prediction Is Deduction but Explanation Is Abduction, *Proceedings IJCAI 89*, pp. 1055-1060.

[Shanahan, 1990] M.P.Shanahan, Representing Continuous Change in the Event Calculus, *Proceedings ECAI 90*, pp. 598-603.

[Shanahan, 1995a] M.P.Shanahan, Default Reasoning about Spatial Occupancy, *Artificial Intelligence*, vol 74 (1995), pp. 147-163.

[Shanahan, 1995b] M.P.Shanahan, A Circumscriptive Calculus of Events, *Artificial Intelligence*, vol 77 (1995), pp. 249-284.

[Shanahan, 1996a] M.P.Shanahan, Robotics and the Common Sense Informatic Situation, *Proceedings ECAI 96*, pp. 684-688.

[Shanahan, 1996b] M.P.Shanahan, Noise and the Common Sense Informatic Situation for a Mobile Robot, *Proceedings AAAI 96*, pp. 1098-1103.

[Shanahan, 1997a] M.P.Shanahan, *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia*, MIT Press (1997).

[Shanahan, 1997b] M.P.Shanahan, Noise, Non-Determinism and Spatial Uncertainty, *Proceedings AAAI 97*, pp. 153-158.

[Shanahan, 1998] M.P.Shanahan, Reinventing Shakey, *Working Notes of the 1998 AAAI Fall Symposium on Cognitive Robotics*, pp. 125-135.

[Strawson, 1966] P.F.Strawson, *The Bounds of Sense*, Methuen (1966).