To appear in *Artificial Intelligence*, special issue on Computational Theories of Interaction and Agency, Agre, P.E. and Rosenschein, S.J. (Eds.).

Indexical Knowledge and Robot Action — A Logical Account

Yves Lespérance and Hector J. Levesque*
Department of Computer Science
University of Toronto
Toronto, ON, M5S 1A4 Canada
{lesperan,hector}@ai.toronto.edu

January 10, 1994

*Fellow of the Canadian Institute for Advanced Research

Abstract

The knowledge required for action is generally indexical rather than objective. For example, a robot that knows the relative position of an object is generally able to go and pick it up; he need not know its absolute position. Agents may have very incomplete knowledge of their situation in terms of what objective facts hold and still be able to achieve their goals. This paper presents a formal theory of knowledge and action, embodied in a modal logic, that handles the distinction between indexical and objective knowledge and allows a proper specification of the knowledge prerequisites and effects of action. Several kinds of robotics situations involving indexical knowledge are formalized within the framework; these examples show how actions can be specified so as to avoid making excessive requirements upon the knowledge of agents.

1 Introduction

1.1 Motivation

Designing autonomous robots or other kinds of agents that interact in sophisticated ways with their environment is hard; you need good tools to do it. Designs should be based on well-developed theories of the interactions that agents have with their environment. There has been a widespread perception that *logic-based formalisms* are unsuitable for this task because classical logic can only represent objective knowledge, it cannot capture the context-relativity, the "situatedness" of action and of the knowledge required for action. In this paper, we will show that this view is inaccurate. We will present a logical theory of knowledge and action that does accommodate indexicality and show that it provides a suitable framework for modeling many kinds of interactions that typically occur between a robot and its environment. Logic-based theories such as ours can help clarify many notions that are used in informal analyses of interaction. No doubt such theories do not capture all aspects of situatedness and much work remains to be done to show that they can be successfully applied to robot design. But we think that after having examined this work, one will agree that it makes an important contribution to the theoretical foundations of the field and has significant potential for applicability, no matter what kind of architecture turns out to be best for achieving reactive yet sophisticated behavior.

A key feature of agent-environment interaction is *incomplete knowledge*: agents typically know very little about their environment. Many theories of action and most existing planners have completely ignored the need to deal with what agents know and need to know by unrealistically assuming that agents always have perfect knowledge of the domain under consideration. But in the past decade, Moore [41, 40], Morgenstern [42, 43], and other researchers have proposed theories of knowledge and action that do address this need.

For example, it is *physically possible* for someone to withdraw funds at a teller machine by inserting a bank card and pressing a sequence of keys (assuming the machine is working, the card is not defective, there is money available, and so on). But this a very weak notion; by the same token, it is physically possible for a monkey to write a sonnet at a typewriter. To say that an agent is truly *able* to withdraw funds, we would want to insist that that the agent *know* among many other things, his bank identification code. The reason this stronger notion of ability is so useful is that it allows us to make sense of actions that otherwise seem to have no relevant effect. For example, how else to explain an agent who always takes a small piece of paper out his wallet and looks at it carefully before pushing the keys on the teller machine?

To understand actions such as these, we need to consider agents that have knowledge about

- the *effects* various actions would have upon their knowledge
- the *knowledge prerequisites* of the actions they might do — the conditions under which they are *able* to achieve goals by doing actions.

Theories such as Moore's and Morgenstern's do provide reasonable answers to the question of what the relationship between knowledge and action is and have considerably advanced our understanding of the issues involved. But many problems remain. This work deals with one aspect of the relationship between knowledge and action that has been neglected and is inadequately handled by these theories: the fact that the knowledge required for action is often relative to the agent's perspective — that it is often *indexical* (or relative) knowledge rather than objective knowledge. For example,

- if a robot knows the relative position of an object he can go and pick it up — knowing the absolute position of the object is neither necessary nor sufficient (he might not know where he is)
- one can soft-boil an egg if one has a timer — knowing at what time it will be done is neither necessary nor sufficient (one may not know what time it is)

In these cases, the agent has very incomplete knowledge of his situation, at least as far as objective facts are concerned, but this does not interfere with his ability to achieve his goals by doing actions. More generally, an agent may have sufficient knowledge to be able to achieve his goals even if he does not know

- where he is
- what time it is
- which objects are around him
- where these objects are located (in absolute terms)
- who he is.

This is the case because the knowledge required for physical interactions with the environment is indexical knowledge, knowledge about how one is related to things in the environment or to events in one's history. This should not come as a surprise, since agents act upon their environment from a particular perspective, a particular place and moment in time. The same action done at different places and times has different effects. So it makes sense that the prerequisites of an action should involve knowledge that is relative to this perspective. Similarly and fortunately, the knowledge supplied by perception is indexical knowledge. For example, by using his sonar, a robot comes to know how far from him an object is (at the current time); he does not learn anything about the object's absolute position. Indexicality has of course been a central theme of the "situated action" paradigm [52, 5, 54]. Several researchers have argued that indexical knowledge plays a major role in the operation of reactive agents [2, 53].

Existing theories of the relationship between knowledge and action cannot handle these cases properly, as they do not accommodate the distinction between indexical knowledge and objective knowledge. They impose unnecessarily strong knowledge requirements upon agents before sanctioning their ability; they tend to require objective knowledge when indexical knowledge would be sufficient. They also cannot represent accurately the effects of action upon knowledge. We need an account of ability that squares with the fact that an agent may be missing very basic facts about his objective situation and that of the things around him and still be capable of achieving his goals. The reason we care about this is that this form of incomplete knowledge is the norm rather than the exception for agents in the world.

This is not to say that objective knowledge is never necessary for action. Because it is independent of the context, it can be stored or communicated without the need for adjustments to compensate for changes in the context. It is clear that human agents have all sorts of objective knowledge about the world and draw upon a variety of external sources of objective knowledge such as textbooks, databases, maps, timetables, etc. In hierarchical planning, it is likely that top-levels plans involve actions that are specified in a relatively objective way (e.g., in planning a car trip across the province, one first decides on what roads to follow and only later on whether to turn right or left). What this means is that a theory of knowledge and action must accommodate both the indexical knowledge involved in interaction with the physical environment and the objective knowledge involved in high-level planning and social interactions. As well, it must allow each kind of knowledge to be mapped into the other provided that knowledge of the context is available, so that objective knowledge can be exploited in physical interactions and indexical knowledge is available for communication, long-term storage, etc.

1.2 The Approach

We want a theory of knowledge and action that handles the distinction between indexical and objective knowledge and accounts for indexical knowledge prerequisites and effects of action. The approach taken is to develop a quantified modal logic that embodies the theory. The logic includes primitives that formalize notions of knowledge, time, action, and historical necessity — the latter notion is used to capture the physical possibility of a course of action for an agent. The notion of ability is formalized as a derived operator (i.e., defined in terms of the notions mentioned above). The epistemic and temporal parts of the logic adequately support the representation of indexical knowledge, keeping it distinct from objective knowledge. We provide a formal semantics for the knowledge operator that handles indexicality; it is a simple and natural extension of the standard possible-world semantic scheme for the logic of knowledge [20, 18]. The complete system permits an adequate characterization of indexical knowledge prerequisites and effects of actions in a wide range of circumstances.

The specification of the theory as a logical system has several advantages. The properties that the theory attributes to agents correspond to provably valid sentences of the logic; this ensures that the theory is precisely specified. Moreover, since attributions of properties to agents emerge as sentences, they can themselves be objects of belief/knowledge by agents; so for example, not only can you state that if an agent does δ he will know whether φ, but also that the agent *knows* that if he does δ he will know whether φ. In fact, our theory models not only interactions but also agents' knowledge about interactions. In it, whenever an agent is able to achieve a goal by doing an action, he knows that this is the case. Whenever a statement like "if agent a knows φ then a can achieve ψ by doing δ" is valid in the logic, then this must be known by all agents. The provision of a formal semantics clarifies the theory's claims and the ontological commitments involved. Finally, the logic can be used to reason formally about knowledge, action, and ability — the kind of reasoning a designer might do in ensuring that the agent being designed is able to achieve a goal. Our methodology differs little from that used by Moore [41, 40], Morgenstern [42, 43], and others who have previously worked on theories of knowledge and action.

Note that our theory is at the "knowledge level" [45], and as such, does not say much about agent architecture. While it uses logic, it does not take any representational stance (i.e., assume that knowledge is represented by sentence-like entities inside agents); this is similar to the position taken by Rosenschein and Kaelbling in their situated automata work [49, 24]. But there is an emerging consensus that satisfactory architectural designs for agents will need to be based on an adequate theory of agent-environment interaction. Our work attempts to provide this kind of foundation.

A substantial part of our efforts has been devoted to the formalization of various application domains within the logical system proposed. This is essential to ensure that the theory is truly useful and that it handles adequately the kind of situations that it is designed for. Moreover, the formalization of actions involves many decisions that have a crucial effect upon what knowledge is required from an agent, but are not settled by the theory; for example, decisions concerning the kind of parameters an action should take. The applications developed show how actions can be specified so as to avoid making excessive requirements upon the knowledge of agents, and how such specifications can be used to prove that an agent is able to achieve a goal by doing an action if he knows certain facts.

Much of our formalization efforts have been directed at a robotics domain, since this kind of application provides the most intuitive examples of situations where indexical knowledge is sufficient for ability. We will describe in detail our formalization of this domain, where a simple robot is involved in various types of interaction with its environment. In our formalization, perception does yield indexical knowledge and ability to act upon an object does not require knowing which object is involved or what its absolute position is. We also show how indexical knowledge and objective knowledge can be and must be related in our framework to deal with the use of maps for navigation. We discuss the representational issues that arise, which have general relevance to the formalization of actions with indexical knowledge prerequisites or effects.

In the next section, we briefly discuss related work. Then in section 3, we give an overview of the logical system that forms the basis of our framework. The core of the paper is section 4, where we describe our formalization of various types of interactions a simple robot has with its environment. Following that, section 5 briefly describes other application domains that have been formalized, in particular cases where temporal knowledge is required for ability. Then in section 6, we argue that the applications examined, in particular the ones involving temporal knowledge, provide convincing evidence that the distinction between indexical and objective knowledge supported by our framework has substantial practical value and that it cannot be done justice within existing accounts of ability. We conclude in section 7 by discussing the contributions and limitations of this work, and suggesting various directions for further research; we discuss how this work might be applied to the design of autonomous robots and other types of agents.

2 Related Work

This survey is selective and often sketchy; for a more complete discussion, see [29].

2.1 Theories of Knowledge and Action

As argued in the introduction, a theory that explains how agents manage to achieve their goals by doing actions, and in particular why they perform knowledge acquisition actions, must account for the effects of action upon knowledge and for the knowledge prerequisites of action; it must include an account of ability as opposed to mere "physical possibility". The most influential work in this area is Moore's theory of knowledge and action [40, 41]. His framework can be described as a combination of first-order dynamic logic (a modal logic of action) [13] with an S4 modal logic of knowledge [19, 18].[1] On the matter of knowledge prerequisites of action, notice that one may know that an action would achieve a goal without knowing how to execute that action; for example, one may know that cooking beef bourguignon would impress the party's guests without knowing how to cook beef bourguignon. For Moore, knowing how to do an action amounts to knowing what primitive actions that action (description) stands for; knowing how to do an action requires having *de re* as opposed to *de dicto* knowledge of the action.

Let us review the philosophical lore on this distinction. *De re* knowledge attributions are said to assert the existence of some kind of epistemic relation between the knower and some entity. The intuition behind such attributions is that they apply to cases where the agent's knowledge is sufficiently precise to pick up a particular entity, as opposed to being about whatever satisfies a description. The latter cases are said to involve mere *de dicto* knowledge. For example, if John is told that someone among his co-workers has been selected to be the new manager, but is not told whom, he may then come to believe that the new manager (whoever he is) must be happy — a *de dicto* rather than *de re* belief. If he is later told that the new manager is Paul, he will then come to have a *de re* belief of Paul that he must be happy. Following Hintikka [19], knowing who/what θ is is usually taken to amount to knowing of some x that it is θ *(de re)*. The question of what precisely is required for an agent to have *de re* knowledge has been the subject of much philosophical debate (see [28] for a discussion). In AI, the common answer has been that having *de re* knowledge of some entity requires knowing a *standard name* for that entity [26, 35], a view

[1]Strictly speaking, the framework is not the modal logic just described, but the encoding in first-order logic of the semantics of this modal logic; thus the part of the logic dealing with action is closely related to the situation calculus [38].

shared by Moore as well as the present work. Since what standard names refer to must be common knowledge; this means that they must be objective, and thus, that *de re* knowledge must in some sense always be objective knowledge. But this can be a bit misleading, as knowing what the relative position of an object is or knowing what primitive actions an action stands for hardly qualify as having objective knowledge. The domain over which one quantifies matters. In such cases, one can at best say that one has "objective" knowledge of a relational property.

So Moore uses this distinction in his formalization of ability. For atomic actions, his account goes as follows: an agent is able to achieve a goal by doing an action if and only if he knows what the given action is, and knows of himself that it is physically possible for him to do the action next and that his doing the action next necessarily results in the goal being achieved. Note that the agent is required to have *de re* knowledge of himself — to know who he is. Complex actions are handled recursively; for instance, an agent is said to be able to achieve a goal by doing a sequence of two actions if and only if by doing the first action, he is able to achieve the goal of being able to achieve the main goal by doing the second action. The agent need not initially know what all the actions that make up his plan are; he needs only know that he will know what to do next at each step of the plan. Note that in most cases, the specification of an action need not say anything explicit about its knowledge prerequisites; these fall out of the theory's general principles and the specification of the conditions under which one knows what the action is.

The requirement that the agent know what the action is has interesting results when the action is an instance of a parametrized procedure (e.g. DIAL(COMBINATION(SAFE1))); in many such cases, agents know what the procedure is and one wants to say that an agent knows what the action is if he knows what the arguments are; it is easy to state this in Moore's formalism (technically, one states that the procedure is an epistemically rigid function). But note that assuming that an instance of a procedure is known whenever one knows what the parameters are is often wrong. For example, whether one knows how to PICKUP(BLOCK1) has nothing to do with whether one knows a standard name for BLOCK1; it has more to do with whether one knows the relative position of BLOCK1, or more generally where BLOCK1 is relative to oneself. One has to be very careful as to how one parametrizes the actions, and in many cases, the parameters must be indexical descriptions.

But Moore's framework does not really accommodate the distinction between indexical and objective knowledge. His logic does not handle knowledge that is indexical with respect to the agent; it does not capture an agent's concept of himself, and knowledge that involves this concept (e.g., an agent's knowing that he himself is holding a block). So the account requires the agent to know who he is (know a standard name for himself, something clearly objective) in order to be judged able to achieve a goal by doing an action. This is clearly unnecessary. What need would a very simple agent, say a robot insect, have for knowing who he is? Because of this, the formalism cannot really model what knowing something under an indexical description amounts to. Yet this seems to be an important feature of how reactive agents deal with their environment (see section 6 for more discussion of this). Moore's logic also does not handle knowledge about absolute times. So for instance, it is not possible to say that after looking at a clock, an agent knows what time it is.[2] These expressive limitations together with the way the account treats parametrized procedures mean that formalizations of various domains in Moore's framework tend to require too much objective knowledge and not enough indexical knowledge.

Let us point out a few other limitations of Moore's account of ability. Firstly, the notion formalized is a highly idealized version of the commonsense notion: one is taken as able to achieve a goal only if one knows that doing the action absolutely guarantees that the goal will be achieved; just knowing that it is highly likely to achieve the goal is not enough. It would be great to have a version of the account that deals with uncertainty, but this is not something we attempt in the present work. Secondly, it may be argued that ability is not just a matter of knowing what actions to take. For example, does knowing how to play a sonata only require knowing which notes to play and does knowing how to ride a bicycle only require knowing which movements to make? On the other hand, it is not clear that these examples point to some essential flaw in the account; perhaps it is just a matter of making explicit the temporal and memory constraints associated with the action (e.g., one needs to recover quickly from playing a note, as well as keep track of what note to play next). Finally, there are cases in commonsense discourse where we individuate actions in terms of their effects rather than in terms of what body movements or effector commands get executed (e.g., one may say that at the end of every working day, a sales representative does the same action no matter where he might be, he goes home). But this is a different sense of "doing an action" from the one we are concerned with; we are not trying to produce a semantic account of natural language discourse about action.

Let us briefly discuss other theories of knowledge and action. One of the unattractive features of the logic of knowledge included in Moore's framework is that knowledge is assumed to be closed under logical consequence. Konolige [25] has developed a theory of knowledge and action, based on an account of knowledge as a predicate on sentences, that avoids this defect.[3] His account of ability is essentially a recasting of Moore's into his framework; only the same restricted class of actions is handled.

A theory that significantly extends this coverage has been developed by Morgenstern [42, 43]. It handles both concurrent actions and plans involving multiple agents. Simpler cases are treated as in Moore's. Her account of knowledge is also syntactic, but differs significantly from Konolige's (she does not use the Tarskian approach to the paradoxes and there is a single truth predicate for all the languages in the hierarchy). Note that knowledge is, in fact, closed under logical consequence in her account. Her argumentation against classical logics of knowledge is not based on the consequential closure problem, but

[2]Technically speaking, Moore models knowledge with an accessibility relation that ranges over (instantaneous) world states. He cannot represent knowledge that is indexical with respect to the agent because there is nothing to characterize who the agent thinks he might be in these states. He also cannot represent knowledge about absolute times because his states have no absolute time is associated with them. This should become clearer after our accounts of knowledge and ability have been introduced.

[3]Note however that such "syntactic" accounts have been claimed to have the opposite defect, that is, to individuate knowledge states *too finely* [36]; for instance, it is far from clear that a belief that φ and φ' is any different from a belief that φ' and φ.

on the claim that they cannot express the weak knowledge prerequisites involved in multi-agent planning. For example, if Paul does not know how to fix a leaky faucet, but knows that his friend John is somehow able to do it, then he is able to fix the faucet by asking John to do it for him. However, recent work by Nunes and Levesque [46] undermines this argument; they shows that the notion of “somehow being able to achieve a goal” can in fact be modeled in a possible-world framework.

Neither Konolige nor Morgenstern recognize the role of indexical knowledge in action; their formalisms have the same limitations as Moore’s in this respect. One researcher who did recognize it is Haas. In [16], he sketches how indexical knowledge might be handled in a specification of ability; but he does not formalize his proposals.

2.2 Theories of Indexical Knowledge

There has been a lot of work on indexical knowledge in philosophy, but we can only mention a few references here (see [29]). Our account of indexical knowledge is inspired from Lewis’s view that having a belief involves ascribing a property to oneself and the current time [37]. In [47], Perry argues convincingly that indexicality is an essential feature of propositional attitudes.

Let us say a bit more about some recent work by Grove [14] and Grove and Halpern [15], where they propose a logic of knowledge that does handle indexicality. Their account of knowledge is quite similar to ours; in fact, their semantics is essentially the same as ours. However, their work is more narrowly focussed than ours; they do not account for action and its relationship to knowledge, and the published version of their work does not even consider time. On the other hand, their work is technically very thorough; they discuss several logical systems with varying degrees of expressiveness, both propositional and first-order, and they provide complete axiomatizations and complexity results.[4] They also develop an appealing approach to quantification into epistemic contexts that permits a resolution of the ambiguity of *de re* reports with respect to the way the knower is referring to the object.

2.3 Reactive Behavior and Situated Action

It has been argued that indexicality plays an important role in how agents manage to react in a timely manner to changing environmental conditions (e.g., avoid collisions with moving objects). Let us look at some recent work on *reactive agents* where this comes out. This is a promising application area for our work.

The classical AI paradigm with respect to the production of intelligent behavior involves a smart planner that searches for a plan that achieves the agent’s goal and a dumb executor that carries out the plan in a mechanical way. In recent years, there has been a definite movement towards exploring alternatives to this paradigm. It is felt that because of the

[4]Interestingly, their results seem to show that the complexity of deciding whether a sentence is valid is no worse in systems that accommodate the distinction between objective and indexical knowledge than in comparable systems that do not.

emphasis it places on search for complete plans, classical deliberative planning is too slow for producing reactive behavior. Moreover, it is not sufficiently grounded in perception; much of the effort expended on constructing a plan may be wasted if environmental conditions change in the meantime. The alternative architectures proposed achieve reactivity by emphasizing environment monitoring and the selection of actions appropriate to conditions; the focus is on developing a sophisticated executor.

Agre and Chapman [2, 3] have been among the most radical in their reevaluation of the classical paradigm. Their views have been influenced by anthropological theories of action [1, 54]. They emphasize the complexity of the real situations in which action occurs, the uncertainty of the information the agent may have about them, and the need for reactivity. This leads them to argue that the production of most activity does not involve the construction and manipulation of explicit representations of the world; the associated computational costs are just too prohibitive. They say that [2, p. 268]:

> Rather than relying on reasoning to intervene between perception and action, we believe activity mostly derives from very simple sorts of machinery interacting with the immediate situation. This machinery exploits regularities in its interaction with the world to engage in complex, apparently planful activity without requiring explicit models of the world.

The architecture they propose involves peripheral modules for perception and effector control and a central system that mediates between the two. They argue that combinational networks, that is, circuits computing logic functions, can form an adequate central system for most activities; thus in such cases, the central system has no state. They have built various application systems to provide support for their analysis. One of these systems, called Pengi, plays a videogame where one controls a penguin that navigates in a maze, pushes away ice blocks, and confronts malicious "killer bees" [2].

From our point of view, the most interesting elements of Agre and Chapman's scheme are their notions of indexical-functional entities and aspects.[5] They claim that traditional domain representations involving objective facts such as "(AT BLOCK-213 427 991)" are inappropriate because they do not make reference to the agent's situation or goals. They see the machinery in their networks as registering indexical-functional entities such as "the block I'm pushing" and indexical-functional aspects such as "the bee I intend to clobber is closer to the projectile than I am". These entities and aspects are indexical because they depend on the agent's situation; they are also functional because they depend on his purposes. Clearly, Agre and Chapman find the notion of indexical information useful in designing their robots and explaining how they behave. However, they do not propose any formal version of this modeling scheme.

Subramanian and Woodfill [53] have recently proposed an interesting account of reactive agent architectures and the role indexical representations play in them; their work includes a computational complexity analysis that attempts to trace the source of the efficiency

[5]This is Agre and Chapman's terminology. There are of course no such things as indexical (or agent-relative) *entities*; but there are indexical (or agent-relative) concepts or notions of things.

gains associated with the use of indexical representations. To model the world as viewed by reactive agents, they use a version of the situation calculus [38] with a vocabulary that includes indexical terms. The logical constant `Now` is used to refer the current situation. Constraints are introduced to ensure that every situation has at most one predecessor and the function `Before` is used to refer to a situation's predecessor (e.g., `Before(Now)` refers to the situation prior to the most recent action). Domain-dependent indexical terms, such as `This-Block`, are also used. Subramanian and Woodfill show how the kind of indexical control rules that determine what a reactive agent will do next (one-step planning) can be specified in this framework. This is also done for plan monitoring rules, which are used to deduce what should be the case after an action has been performed. Their complexity analysis of this kind of setting traces the efficiency gains associated with the use of indexical representations to the fact that indexical theories can be propositionalized without the usual combinatorial explosion because they do not quantify over all entities of the relevant type; instead, they refer to the entities involved indexically. In their words: "the power of using these indexical terms is that it gives us implicit quantification."

Independently of whether this analysis is correct, it must be noted that Subramanian and Woodfill's framework inherits most of the limitations of the ordinary situation calculus, in particular, its inability to handle knowledge acquisition actions. Moreover, the semantics fails to distinguish between the context dependence of indexical terms and the simple world-dependence of non-rigid constants. Finally, the theory does not account for the distinctive logical behavior of indexical terms, for example, the fact that the past is determined while the future is not.

Another conception of how reactive behavior can be produced has been proposed by Rosenschein and Kaelbling [49, 24]. Their approach is of particular interest to us because a logic is used to specify reactive agents and the environments in which they operate. Design tools based on the logic are also developed; these tools facilitate the specification of complex agents and allow high-level specifications to be "compiled" into circuit-level ones.

The architecture they propose for reactive agents [23] also avoids formal manipulation of explicit representations. It involves a perceptual component and an action selection component, both of which may have state (registers). They also argue that the lack of explicit representations does not mean that one loses the ability to ascribe semantic content to machine states. They propose an alternative way of doing this — the situated automata view. It involves viewing the agent and its environment as coupled automata. A machine state contains the information that φ (i.e., in that state, the machine knows that φ) if given the state transition function of the machine and world, the fact that the machine is in that state implies that φ must be the case. For example, the fact that a certain wire in a robot carries a "1" would mean that there is an object within a certain radius of his position, if given the state transition function, the wire can only be carrying a "1" if there is an object within that radius. Thus the information content of a machine state is defined in terms of how it is correlated to environmental conditions.

Rosenschein and Kaelbling do not discuss the role of indexical knowledge in the production of reactive behavior. Their logic suffers from the same kind of expressive limitations as Moore's with respect to indexicality; it does not handle knowledge that is indexical with respect to the agent, and while it accommodates temporally indexical knowledge, it does not handle temporally objective knowledge. This may come as a surprise because their formalization of examples such as the one involving a robot that keeps track of whether a moving object is within shouting distance in [49] uses many indexical-sounding propositions. But their logic does not really model all of that indexicality. When indexical-sounding domain-dependent symbols are used, it's easy to lose track of what the formalism really handles. However, their work strongly suggests that a theory of knowledge and action that handles the distinction between indexical and objective knowledge would be very useful for producing better accounts of reactive behavior and better tools for the design of reactive agents.

3 A Logic of Indexical Knowledge, Action, and Ability

In this section, we briefly review a formal theory of indexical knowledge, action and ability, and some of its properties. A much more detailed examination of the theory, including a discussion of the general issues involved can be found in [29, 33]. Here, we simply present the theory in sufficient detail to underwrite the applications of the next two sections.

Our theory is formulated in a many-sorted first-order modal logic with equality. We assume familiarity with conventional first-order logic as in [39], and at least some acquaintance with the standard apparatus of modal logics, as described in [21] or [8].

3.1 Syntax

We want to be able to express attributions of indexical knowledge in our logic, for example, that Rob knows that he himself was holding a cup five minutes ago. In such cases, what is known is a "proposition" that is relative. It may be relative to the knower, or to the time of the knowing, or perhaps to other aspects of the context. To handle this, our language includes two special terms: **self**, which denotes the current agent, and **now**, which denotes the current time; these terms are called *primitive indexicals*. Non-logical (domain-dependent) symbols may also depend on the current agent and time for their interpretation, for example, $\exists x \textsc{Holding}(x)$ may express the fact that the current agent is currently holding something — we say that such symbols are non-primitive indexicals. Our semantics handles this by interpreting terms and formulas with respect to *indices*, which consist of a possible-world, modeling the objective circumstances, and an agent and time, modeling the context. Note that **self** and **now** are *not* intended to be formal counterparts of similar sounding English words and behave differently from any such words (more about this shortly).

The language we use is called *LIKA* and as any first-order logical language, it divides syntactically into terms and formulas.[6] The terms here, however, are of four different

[6] *LIKA* stands for "Language of Indexical Knowledge and Action". In [29], we used the name $\mathcal{L}_{index}$ instead of *LIKA*.

sorts: terms for ordinary individuals (as usual), temporal terms, agent terms, and action terms. For each of these four sorts, there are both variables and function symbols (that is, functions whose values will be of the proper sort) and as usual, constants are taken to be 0-ary function symbols. We will use the metavariables v, F, and θ to range over variables, function symbols, and terms respectively, with superscripts i, t, a, and d used to indicate the sort: individual, temporal, agent, and action respectively. So, for example, v^t stands for a temporal variable, and θ^d stands for an action term. Syntactically, terms are formed in the obvious way, with the following restrictions: firstly, only temporal function symbols may take temporal terms as arguments, and secondly, action terms cannot appear as arguments to function symbols of any sort.

The atomic formulas include predications using predicate symbols and terms, written $R(\theta_1, \ldots \theta_n)$, which are used to assert that $\theta_1, \ldots \theta_n$ stand in static relation R at the current time and for the current agent. We also have equality expressions $(\theta_1 = \theta_2)$, between terms of the same sort, as well as expressions of temporal precedence $(\theta_1^t < \theta_2^t)$. Finally, $\mathbf{Does}(\theta^d, \theta^t)$ is used to assert that the current agent does action θ^d starting from the current time and ending at time θ^t.

For non-atomic formulas, we have negations, implications, and universal quantifications, and all the standard abbreviations (such as disjunctions; see below). Finally, if φ is a formula, then so are $\mathbf{At}(\theta^t, \varphi)$, $\mathbf{By}(\theta^a, \varphi)$, $\mathbf{Know}(\varphi)$, and $\Box\varphi$. $\mathbf{At}(\theta^t, \varphi)$ means that φ holds at time θ^t, that is, when θ^t is taken to be the current time. $\mathbf{By}(\theta^a, \varphi)$ means that φ holds when θ^a is taken to be the current agent. $\mathbf{Know}(\varphi)$ is used to say that the current agent knows at the current time that φ. If φ contains indexicals, $\mathbf{Know}(\varphi)$ should be taken as attributing indexical knowledge — knowledge the agent has about himself and the current time. For example, $\mathbf{Know}(\exists x \textsc{Holding}(x))$ could mean that the agent knows that he himself is currently holding something. Finally, $\Box\varphi$ is used to say that φ is historically necessary, that is, that it must hold, given everything that has happened up to now. This completes the syntactic specification of the language.

The notions of free variable, bound variable, and term that is free for a variable in a formula are assumed to have their usual definition. We use $\varphi\{v \mapsto \theta\}$ to stand for the result of substituting θ for all free occurrences of v in φ, provided that v and θ belong to the same sort.

All symbols of the language other than the function symbols and the predicate symbols are considered to be *logical* symbols, and will have a fixed interpretation. In the examples of the next section, we will introduce domain-dependent function or predicate symbols (with names like $\textsc{pos}$ or $\textsc{Holding}$) as well as axioms governing their interpretation.

3.2 Semantic Structures

The terms and formulas of *LIKA* are understood in terms of the following semantic components: $\mathcal{A}$, $\mathcal{I}$, $\mathcal{T}$, $\mathcal{D}$, $\mathcal{W}$, Φ, $\prec$, $\approx$, Δ, and K. Instead of a single domain of discourse, we have non-empty domains for each sort: a set $\mathcal{A}$ of agents, a set $\mathcal{I} \supseteq \mathcal{A}$ of individuals, a set $\mathcal{T}$ of time points, and a set $\mathcal{D}$ of primitive actions. The terms of *LIKA* are taken to refer to elements of these domains, and quantification is handled in the usual way.

$\mathcal{W}$ is a set of temporally extended possible worlds. As explained earlier, we handle indexicals by interpreting terms and formulas with respect to *indices*, which are triples consisting of a world, an agent, and a time; so $\mathcal{E} = \mathcal{W} \times \mathcal{A} \times \mathcal{T}$ is the set of indices. We take $\mathsf{a}, \mathsf{i}, \mathsf{t}, \mathsf{d}, \mathsf{w}$, and e (possibly subscripted, primed, etc.), as ranging over arbitrary elements of $\mathcal{A}, \mathcal{I}, \mathcal{T}, \mathcal{D}, \mathcal{W}$, and $\mathcal{E}$ respectively. The **By** and **At** operators are used to change the agent and time component of the index respectively. The denotations of terms and the truth of predications are handled in the obvious way using a function Φ which, at each index, maps function symbols to functions over the appropriate domains and predicate symbols to relations over the appropriate domains.

Turning now to the temporal aspects, formulas containing $<$ are understood in terms of $\prec$, which is a relation over $\mathcal{T}$ whose intended interpretation is "is earlier than." $\approx$ is a family of accessibility relations — one for each time point — that is used to interpret the historical necessity operator $\Box$. Intuitively, $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$ if and only if w and w^* differ only in what happens after time t. Formulas containing **Does** are interpreted in terms of $\Delta \subseteq \mathcal{D} \times \mathcal{E} \times \mathcal{T}$, which determines which actions are done by which agents in which worlds over which time intervals: $\langle \mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}_e\rangle \in \Delta$ if and only if action d is done by agent a from time t_s to time t_e in world w.

Finally, our semantics for knowledge is a simple generalization of the standard possible-world scheme [27, 20]. The knowledge accessibility relation K is taken to hold over *indices* rather than plain possible worlds: $\mathsf{K} \subseteq \mathcal{E}^2$. Informally, $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle\rangle \in \mathsf{K}$ if and only if as far as agent a at time t in world w knows, it may be the case that w' is the way the world actually is *and* he is a' *and* the current time is t'. Thus, we model the knowledge state of an agent at a time in a world by a set of indices, which characterizes not only which worlds are compatible with what the agent knows, but also which points of view upon these worlds are compatible with what he knows. In other words, we allow an agent to be uncertain not only about what world he is in, but also about who he is and what time it is.

3.3 Denotation and Satisfaction

More formally, we have the following: a semantic structure M is a tuple

$$\langle \mathcal{A}, \mathcal{I}, \mathcal{T}, \mathcal{D}, \mathcal{W}, \prec, \mathsf{K}, \approx, \Phi, \Delta\rangle.$$

An assignment is a function that maps variables into elements of the domain appropriate to them. $\mathsf{g}\{v \mapsto \mathsf{x}\}$ is the assignment that is identical to g except that it maps variable v into the entity x.

The *denotation* of a term θ in a structure M at an index $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle$ under an assignment g, written $[\![\theta]\!]^{\mathsf{M}}_{\mathsf{e},\mathsf{g}}$ is defined as follows (from now on, we omit the structure under consideration when it is clear from context):

$[\![v]\!]_{\mathsf{e},\mathsf{g}} = \mathsf{g}(v)$.

$[\![\mathbf{self}]\!]_{\mathsf{e},\mathsf{g}} = \mathsf{a}$

$[\![\mathbf{now}]\!]_{\mathsf{e},\mathsf{g}} = \mathsf{t}$

$[\![F(\theta_1, \dots, \theta_n)]\!]_{\mathsf{e,g}} = \Phi(F, \mathsf{e})([\![\theta_1]\!]_{\mathsf{e,g}}, \dots, [\![\theta_n]\!]_{\mathsf{e,g}})$

We can now define what it means for a formula φ to be *satisfied* by a structure M, an index $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t} \rangle$, and an assignment $\mathbf{g}$, which we write $\mathsf{M}, \mathsf{e}, \mathbf{g} \models \varphi$:

$\mathsf{e}, \mathbf{g} \models R(\theta^i_1, \dots, \theta^i_n)$ iff $\langle [\![\theta^i_1]\!]_{\mathsf{e,g}}, \dots, [\![\theta^i_n]\!]_{\mathsf{e,g}} \rangle \in \Phi(R, \mathsf{e})$

$\mathsf{e}, \mathbf{g} \models \mathbf{Does}(\theta^d, \theta^t)$ iff $\Delta([\![\theta^d]\!]_{\mathsf{e,g}}, \mathsf{e}, [\![\theta^t]\!]_{\mathsf{e,g}})$

$\mathsf{e}, \mathbf{g} \models \theta_1 = \theta_2$ iff $[\![\theta_1]\!]_{\mathsf{e,g}} = [\![\theta_2]\!]_{\mathsf{e,g}}$

$\mathsf{e}, \mathbf{g} \models \theta^t_1 < \theta^t_2$ iff $[\![\theta^t_1]\!]_{\mathsf{e,g}} \prec [\![\theta^t_2]\!]_{\mathsf{e,g}}$

$\mathsf{e}, \mathbf{g} \models \neg\varphi$ iff it is not the case that $\mathsf{e}, \mathbf{g} \models \varphi$

$\mathsf{e}, \mathbf{g} \models (\varphi_1 \supset \varphi_2)$ iff either it is not the case that $\mathsf{e}, \mathbf{g} \models \varphi_1$, or $\mathsf{e}, \mathbf{g} \models \varphi_2$

$\mathsf{e}, \mathbf{g} \models \forall v \varphi$ iff for every entity x in the domain appropriate to v, $\mathsf{e}, \mathbf{g}\{v \mapsto \mathsf{x}\} \models \varphi$

$\mathsf{e}, \mathbf{g} \models \mathbf{At}(\theta^t, \varphi)$ iff $\langle \mathsf{w}, \mathsf{a}, [\![\theta^t]\!]_{\mathsf{e,g}} \rangle, \mathbf{g} \models \varphi$

$\mathsf{e}, \mathbf{g} \models \mathbf{By}(\theta^a, \varphi)$ iff $\langle \mathsf{w}, [\![\theta^a]\!]_{\mathsf{e,g}}, \mathsf{t} \rangle, \mathbf{g} \models \varphi$

$\mathsf{e}, \mathbf{g} \models \mathbf{Know}(\varphi)$ iff for all e', such that $\langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$, $\mathsf{e}', \mathbf{g} \models \varphi$

$\mathsf{e}, \mathbf{g} \models \Box\varphi$ iff for all w^* such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$, $\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t} \rangle, \mathbf{g} \models \varphi$

A formula φ is *satisfiable* if there exists a structure M, index e, and assignment $\mathbf{g}$, such that $\mathsf{M}, \mathsf{e}, \mathbf{g} \models \varphi$. A formula φ is *valid* (written $\models \varphi$) if it is satisfied by all structures, indices, and assignments.

As mentioned earlier, our logic is not intended to be a formalization of the behavior of English indexicals; we see it as a specification language, a tool for modeling how agents interact with their environment. In English, there are true indexicals (I, you, now, here, etc.), which refer to aspects of the utterance context no matter where they appear, and there are quasi-indexicals/quasi-indicators [7] (I myself, you yourself, he himself, etc.), which are used to report that an agent has an indexical mental state. The behavior of our *primitive indexicals* **self** and **now** displays characteristics of both categories. When **self** occurs outside the scope of **Know** or **By**, it behaves like the English indexical 'I' and when **now** occurs outside the scope of **Know** or **At**, it behaves like the English indexical 'now'. In the scope of **Know** on the other hand, **self** and **now** behave like quasi-indexicals — there are no temporal quasi-indexicals in English, but one can imagine how a temporal analogue of 'he himself' would work. Finally, when **self** occurs in the scope of **By** and when **now** occurs in the scope of **At**, they behave like pronouns that are bound by the operator; this is similar to the way in which an expression like 'the day before' depends on surrounding temporal operators as in 'tomorrow, it will be the case that the day before, I wrote this line'. We wanted to keep the logic simple and allow typical assertions about knowledge and action to be expressed concisely. This led to the chosen design. These features come at the cost of a simple relationship to English. But we feel that the logical behavior of our primitive indexicals is easily understood once one realizes it is different from that of any English analogues. This is not to say that formalizing the behavior of English indexicals is uninteresting. In fact, we think that our framework can be used as a foundation to build a model that relates the context sensitivity of language to that of mental states and action. We return to this topic in section 7.2.

3.4 Constraints

To ensure that the semantics adequately models the notions that we are trying to capture (knowledge, historical necessity, etc.), we impose various constraints on semantic structures. When we speak of a valid or satisfiable formula we always mean relative to semantic structures that satisfy these constraints. Here we will simply list the constraints without justification (and minimal explanation):

- (S4 epistemic logic)
  K must be reflexive and transitive.
- (linear time logic)
  $\prec$ must be a strict total order, that is, transitive, connected, and irreflexive.[7]
- (start time is before end time)
  If $\langle \mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}_e\rangle \in \Delta$, then $\mathsf{t}_s \preceq \mathsf{t}_e$.
- (historical necessity is an S5 operator)
  For all $\mathsf{t} \in \mathcal{T}, \approx_{\mathsf{t}}$ must be an equivalence relation.
- (possibilities do not increase over time)
  If $\mathsf{w} \approx_{\mathsf{t}_2} \mathsf{w}^*$ and $\mathsf{t}_1 \preceq \mathsf{t}_2$, then $\mathsf{w} \approx_{\mathsf{t}_1} \mathsf{w}^*$.
- (equivalent worlds support the same basic facts and knowledge)
  If $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$, then, letting $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle$ and $\mathsf{e}^* = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}\rangle$, it must be the case that
  1. for any predicate R, $\Phi(R, \mathsf{e}^*) = \Phi(R, \mathsf{e})$,
  2. for any function symbol F, $\Phi(F, \mathsf{e}^*) = \Phi(F, \mathsf{e})$,
  3. for any e', $\langle \mathsf{e}^*, \mathsf{e}'\rangle \in \mathsf{K}$ iff $\langle \mathsf{e}, \mathsf{e}'\rangle \in \mathsf{K}$.
- (equivalent worlds support the same actions)
  1. If $\mathsf{w} \approx_{\mathsf{t}_e} \mathsf{w}^*$, then $\Delta(\mathsf{d}, \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}_e)$ iff $\Delta(\mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}_e)$,
  2. if $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$ and $\mathsf{t}_s \prec \mathsf{t}$, then
     $$\begin{gathered}\text{there exists } \mathsf{t}_e \text{ such that } \mathsf{t} \prec \mathsf{t}_e \text{ and } \Delta(\mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}_e)\\ \text{if and only if}\\ \text{there exists } \mathsf{t}'_e \text{ such that } \mathsf{t} \prec \mathsf{t}'_e \text{ and } \Delta(\mathsf{d}, \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{t}'_e).\end{gathered}$$

[7] A relation R is connected if and only if for any x_1 and x_2, either $\mathsf{x}_1 \mathsf{R} \mathsf{x}_2$ or $\mathsf{x}_1 = \mathsf{x}_2$ or $\mathsf{x}_2 \mathsf{R} \mathsf{x}_1$; a relation R is irreflexive if and only if for no x is it the case that $\mathsf{x}\mathsf{R}\mathsf{x}$.

- (persistent memory and awareness of actions)
  If $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'\rangle\rangle \in \mathsf{K}$ and $\mathsf{t}_p \preceq \mathsf{t}$, then there exists a time t'_p, where $\mathsf{t}'_p \preceq \mathsf{t}'$, such that $\langle\langle \mathsf{w}, \mathsf{a}, \mathsf{t}_p\rangle, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_p\rangle\rangle \in \mathsf{K}$ and if $\Delta(\mathsf{d}, \langle \mathsf{w}, \mathsf{a}, \mathsf{t}_p\rangle, \mathsf{t})$ then $\Delta(\mathsf{d}, \langle \mathsf{w}', \mathsf{a}', \mathsf{t}'_p\rangle, \mathsf{t}')$.

See [33] for discussion of these constraints.

3.5 Abbreviations

To simplify writing formulas in *LIKA*, it is convenient to use certain abbreviations or notational conventions. Firstly, we assume the usual definitions for $\vee, \wedge, \equiv, \exists, \neq, >, \leq$, and $\geq$. We also introduce a dual to $\Box$: $\Diamond\varphi \stackrel{\text{def}}{=} \neg\Box\neg\varphi$. Next, using the operator **By**, we define a more common version of **Know** that specifies which agent knows the given proposition; similar definitions are assumed for **Does** and other operators that have yet to be introduced (**Res**, **Can**, etc.):

$$\textbf{Know}(\theta^a, \varphi) \stackrel{\text{def}}{=} \textbf{By}(\theta^a, \textbf{Know}(\varphi)), \text{and similarly for } \textbf{Does}, \textbf{Res}, \textbf{Can}, \text{etc.}$$

We have also developed a set of definitions that make it easy to state the occurrence of a large class of complex actions. We first define a new syntactic category, that of *action expressions*. We use the syntactic variable δ to represent members of this category. The category is defined by the following BNF rule:

$$\delta ::= \theta^d \mid \textbf{noOp} \mid (\delta_1; \delta_2) \mid \textbf{if}(\varphi, \delta_1, \delta_2)$$

It includes action terms, which represent simple actions, the **noOp** action which takes no time and changes nothing, $(\delta_1; \delta_2)$, which represents the sequential composition of the actions δ_1 and δ_2, and $\textbf{if}(\varphi, \delta_1, \delta_2)$, which represents the action that consists in doing action δ_1 if the condition φ holds, and in doing action δ_2 otherwise.

We inductively extend **Does** to take action expression arguments as follows:

- $\textbf{Does}(\textbf{noOp}, \theta^t) \stackrel{\text{def}}{=} (\theta^t = \textbf{now})$
- $\textbf{Does}((\delta_1; \delta_2), \theta^t) \stackrel{\text{def}}{=} \exists v_i^t(\textbf{Does}(\delta_1, v_i^t) \wedge \exists v_e^t(v_e^t = \theta^t \wedge \textbf{At}(v_i^t, \textbf{Does}(\delta_2, v_e^t))))$, provided that v_i^t and v_e^t are distinct and do not occur anywhere in θ^t, δ_1, or δ_2
- $\textbf{Does}(\textbf{if}(\varphi, \delta_1, \delta_2), \theta^t) \stackrel{\text{def}}{=} (\varphi \wedge \textbf{Does}(\delta_1, \theta^t)) \vee (\neg\varphi \wedge \textbf{Does}(\delta_2, \theta^t))$

We also introduce the following abbreviations for action expressions:

- $\delta^{\mathsf{k}} \stackrel{\text{def}}{=} \begin{cases} \textbf{noOp} & \text{if } \mathsf{k} = 0 \\ (\delta; \delta^{\mathsf{k}-1}) & \text{if } \mathsf{k} > 0 \end{cases}$
- $\textbf{ifThen}(\varphi, \delta) \stackrel{\text{def}}{=} \textbf{if}(\varphi, \delta, \textbf{noOp})$
- $\textbf{while}_{\mathsf{k}}(\varphi, \delta) \stackrel{\text{def}}{=} \begin{cases} \textbf{noOp} & \text{if } \mathsf{k} = 0 \\ \textbf{ifThen}(\varphi, (\delta; \textbf{while}_{\mathsf{k}-1}(\varphi, \delta))) & \text{if } \mathsf{k} > 0 \end{cases}$

Note that the numbers $\mathsf{k}(\in \mathbb{N})$ in the above are not part of the language, and so cannot be quantified over. Also the last abbreviation above is a bounded form of "while loop"; properly capturing an unbounded form is problematic in the language as it stands.

Let us also define some dynamic-logic-style operators that will be used in our formalization of ability.[8] $\mathbf{AfterNec}(\delta, \varphi)$, which is intended to mean "φ must hold after δ", is defined inductively as follows:

- $\mathbf{AfterNec}(\theta^d, \varphi) \stackrel{\text{def}}{=} \Box\forall v^t(\mathbf{Does}(\theta^d, v^t) \supset \mathbf{At}(v^t, \varphi))$, where v^t is a temporal variable that does not occur free in φ
- $\mathbf{AfterNec}(\mathbf{noOp}, \varphi) \stackrel{\text{def}}{=} \varphi$
- $\mathbf{AfterNec}((\delta_1; \delta_2), \varphi) \stackrel{\text{def}}{=} \mathbf{AfterNec}(\delta_1, \mathbf{AfterNec}(\delta_2, \varphi))$
- $\mathbf{AfterNec}(\mathbf{if}(\varphi_c, \delta_1, \delta_2), \varphi) \stackrel{\text{def}}{=} (\varphi_c \supset \mathbf{AfterNec}(\delta_1, \varphi)) \wedge (\neg\varphi_c \supset \mathbf{AfterNec}(\delta_2, \varphi))$

In terms of this, we define $\mathbf{PhyPoss}(\delta)$, which is intended to mean that it is "physically possible" for **self** to do action δ next (even though he may not be able to it because he does not know what primitive actions δ stands for), and $\mathbf{Res}(\delta, \varphi)$, read as "$\delta$ results in φ", which means that δ is physically possible and φ must hold after δ:

- $\mathbf{PhyPoss}(\delta) \stackrel{\text{def}}{=} \neg\mathbf{AfterNec}(\delta, \mathbf{False})$
- $\mathbf{Res}(\delta, \varphi) \stackrel{\text{def}}{=} \mathbf{PhyPoss}(\delta) \wedge \mathbf{AfterNec}(\delta, \varphi)$

True (**False**) stands for some tautology (contradiction).

Let us also define a few additional operators:

- $\mathbf{Kwhether}(\varphi) \stackrel{\text{def}}{=} \mathbf{Know}(\varphi) \vee \mathbf{Know}(\neg\varphi)$
- $\mathbf{DoneWhen}(\delta, \varphi) \stackrel{\text{def}}{=} \exists v_s^t \exists v_e^t (v_e^t = \mathbf{now} \wedge \mathbf{At}(v_s^t, \mathbf{Does}(\delta, v_e^t) \wedge \varphi))$, provided that v_s^t and v_e^t are distinct and do not occur anywhere in φ or δ
- $\mathbf{SomePast}(\varphi) \stackrel{\text{def}}{=} \exists v^t(v^t < \mathbf{now} \wedge \mathbf{At}(v^t, \varphi))$, where v^t does not occur free in φ

$\mathbf{Kwhether}(\varphi)$ means that **self** knows whether φ holds; $\mathbf{DoneWhen}(\delta, \varphi)$ means that **self** has just done δ and that φ was true when he started; and $\mathbf{SomePast}(\varphi)$ means that φ held at some point in the past.

[8]These definitions are a bit different from the ones given in [29, 31]. The ones given here make the operators behave exactly as their dynamic logic [13] analogues. The differences are discussed in [33].

3.6 Ability

We base our formalization of ability on that of Moore [40], which in spite of its relative simplicity, does get at the essential connection between the ability of agents to achieve goals and the knowledge they have about relevant actions. It is simpler than his because we do not attempt to handle indefinite iteration (while-loop actions). Moore's formalization of this case is actually defective because it does not require the agent to know that the action will eventually terminate. We leave this case for future research.

Since we are not treating indefinite iteration, we can simply define ability in terms of the other constructs of the logic as follows:

- $\mathbf{Can}(\delta, \varphi) \stackrel{\text{def}}{=} \mathbf{CanDo}(\delta) \wedge \mathbf{Know}(\mathbf{AfterNec}(\delta, \varphi))$
- $\mathbf{CanDo}(\theta^d) \stackrel{\text{def}}{=} \exists v^d\, \mathbf{Know}(v^d = \theta^d) \wedge \mathbf{Know}(\mathbf{PhyPoss}(\theta^d))$, where action variable v^d does not occur free in θ^d
- $\mathbf{CanDo}(\mathbf{noOp}) \stackrel{\text{def}}{=} \mathbf{True}$
- $\mathbf{CanDo}(\delta_1; \delta_2) \stackrel{\text{def}}{=} \mathbf{CanDo}(\delta_1) \wedge \mathbf{Know}(\mathbf{AfterNec}(\delta_1, \mathbf{CanDo}(\delta_2)))$
- $\mathbf{CanDo}(\mathbf{if}(\varphi, \delta_1, \delta_2)) \stackrel{\text{def}}{=} (\mathbf{Know}(\varphi) \wedge \mathbf{CanDo}(\delta_1)) \vee (\mathbf{Know}(\neg\varphi) \wedge \mathbf{CanDo}(\delta_2))$

The definition says that the agent is able to achieve the goal φ by doing action δ, formally $\mathbf{Can}(\delta, \varphi)$, if and only if he can do action δ and knows that after doing δ, the goal φ must hold. $\mathbf{CanDo}(\delta)$ is defined inductively. The first case takes care of simple actions — actions that are represented by action terms. It says that **self** can do a simple action θ^d if and only if he knows what that action is and knows that it is physically possible for him to do it. Note that the definition involves quantifying-in over the class of primitive actions (e.g. "send grasping signal to hand"), as opposed to arbitrary action descriptions. The second case handles the **noOp** action, which trivially holds. The third case handles sequentially composed actions: **self** can do $(\delta_1; \delta_2)$ if and only if he can do δ_1 and knows that after doing it he will be able to do δ_2. The final case takes care of conditional actions: **self** can do $\mathbf{if}(\varphi_c, \delta_1, \delta_2)$ if and only if he either knows that the condition φ_c holds and can do δ_1, or knows that it does not hold and can do δ_2.[9]

Note that we eliminate Moore's requirement that the agent know who he is; instead, we require indexical knowledge. Thus, in the simple action case we require that the agent know what the action is, that he know that it is physically possible for *himself* to do it, and that he know that if *he himself* does it, the goal will necessarily hold afterwards. Mere *de re* knowledge is neither necessary nor sufficient for being able to achieve a goal; we discuss this further and give a concrete example in section 4.2. Also, as discussed in section 5, the fact that our account of ability is based on a more expressive temporal logic has important

[9] This way of defining **Can** is preferable to the one in [29, 31] as it separates the knowledge prerequisites involving the goal from the rest; this makes it easier to prove results involving complex actions see [33] for further discussion.

advantages when dealing with actions whose prerequisites or effects involve knowledge of absolute times and knowing what time it is.

3.7 Properties of the Logic

In this subsection, we list some properties of the logic of *LIKA* that are used in the proofs of the robotics applications of the next section. We show that the logic indeed satisfies these properties in [29, 33], where we also discuss their significance.

The basis of our logic, that is, the part concerned with first-order logic with equality, is standard (the axiomatization in [39] can be used) with one exception: the "axiom" of specialization is restricted to prevent non-rigid terms from being substituted into modal contexts. This yields the following proposition:

Proposition 3.1 (Specialization)

$\models \forall v\varphi \supset \varphi\{v \mapsto \theta\}$, provided that θ is free for v in φ, no occurrence of a function symbol gets substituted into the scope of a **Know**, **At**, or **By** operator, no occurrence of **self** gets substituted into the scope of **Know** or **By**, and no occurrence of **now** gets substituted into the scope of **Know** or **At**.

Knowledge obeys the following principles in our logic:

Proposition 3.2 $\models \mathbf{Know}(\varphi_1 \supset \varphi_2) \supset (\mathbf{Know}(\varphi_1) \supset \mathbf{Know}(\varphi_2))$

Proposition 3.3 If $\models \varphi$, then $\models \mathbf{Know}(\varphi)$

Proposition 3.4 $\models \mathbf{Know}(\varphi) \supset \varphi$

Proposition 3.5 $\models \mathbf{Know}(\varphi) \supset \mathbf{Know}(\mathbf{Know}(\varphi))$

Proposition 3.6 $\models \forall v\, \mathbf{Know}(\varphi) \supset \mathbf{Know}(\forall v\varphi)$

Note that our handling of indexical knowledge affects how the above statements should be read. For instance, proposition 3.5 says that if the agent knows that φ, then he knows that *he himself currently* knows that φ; $\forall a(\mathbf{Know}(a,\varphi) \supset \mathbf{Know}(a, \mathbf{Know}(a,\varphi)))$ is not valid.

We say that a formula φ is *future-blind* if and only if φ contains no occurrences of the **At** operator or the **Does** predicate outside the scope of a **Know** operator except in forms that can be represented as **SomePast** and **DoneWhen**. The following proposition says that a future-blind formula (i.e., one that does not talk about the future) is historically necessary if and only if it is true:

Proposition 3.7 $\models \Box\varphi \equiv \varphi$, provided that φ is future-blind

Finally, we will need the following properties of **Can**:

Proposition 3.8 If $\models \varphi_i \supset \mathbf{Can}(\delta_2, \varphi_e)$, then $\models \mathbf{Can}(\delta_1, \varphi_i) \supset \mathbf{Can}((\delta_1;\delta_2), \varphi_e)$

Proposition 3.9 $\models \mathbf{Can}(\delta, \varphi) \supset \mathbf{Can}(\delta, \mathbf{Know}(\varphi))$

Proposition 3.10
$\models \mathbf{Can}(\mathbf{if}(\varphi_c, \delta_1, \delta_2), \varphi_g) \equiv (\mathbf{Know}(\varphi_c) \wedge \mathbf{Can}(\delta_1, \varphi_g)) \vee (\mathbf{Know}(\neg\varphi_c) \wedge \mathbf{Can}(\delta_2, \varphi_g))$

4 Formalizing a Simple Robotics Domain

In the previous section, we briefly reviewed a theory of indexical knowledge, action, and ability (described in detail in [33]). We claim that this theory forms an adequate framework for the formalization of actions involving indexical knowledge prerequisites or effects. Let us now substantiate this claim. We will formalize a robotics domain within the theory and prove that a robot is able to achieve certain goals by doing certain actions provided that he knows various facts. We will argue that our framework allows a much more accurate modeling of these cases than frameworks that ignore indexicality.

As will become clear, the framework we provide does not turn the formalization of actions that may have indexical knowledge prerequisites or effects into a trivial task. Many decisions must still be made in developing a formalization that have a crucial bearing on its adequacy from the point of view of how much and what kind of knowledge it requires from an agent; for example, decisions as to which parameters a procedure should take. What we provide on this front is some general advice on what to watch for, as well as the exemplary value of the situations that we formalize. Our central admonition in this respect is that *one should be careful not to impose excessive knowledge requirements upon agents in formalizing actions; in particular, one should merely require indexical knowledge, as opposed to objective knowledge, when that is sufficient.* Together with our formalization of the examples, we provide a discussion of how this guideline has been put into practice.

4.1 The Domain

Our domain involves a robot, call him ROB, that moves about on a two-dimensional grid. Since our purpose is not to model situations where multiple agents interact, but to present and justify our account of indexical knowledge and action, our formalization will be based on the assumption that the robot is the only source of activity in the domain. We take our robot to have the following repertory of basic actions (primitives of his architecture): he may move forward by one square, he may turn right or left 90°, he may sense whether an object is on the square where he is currently positioned and if there is one, what shape it has, and he may pick up an object from the current square or put down the object he is holding on the current square. It should be clear that in spite of the simplicity of this domain, it contains analogues to a large number of problems faced by real robotic agents. For instance, one can view objects of particular shapes as landmarks and the robot can then navigate by recognizing such landmarks. We assume that there are no physical obstacles to the robot's movements; in particular, an object being on a square does not prevent the robot from being on it too (one can imagine the robot as standing over the object). Finally, note that the formalization does not model the uncertainty involved in predicting the effects of actions and acquiring knowledge by sensing; this limitation would have to be addressed for the framework to be applicable to real robotics problems.

4.2 Ability to Manipulate an Object

The indexicality of action manifests itself in many ways in this domain. One key way is that a robot can act upon (manipulate) an object if he knows where that object is *relative to himself*; he need not know either the object's absolute position or his own. First consider a simple instance of this where the robot wants to pick up an object and is actually positioned where that object is. Relevant aspects of the domain are formalized by making various assumptions,[10] most of which have to do with the types of action involved. The following assumption specifies the effects of the action PICKUP:

Assumption 4.1 (Effects of PICKUP)

$$\models \forall x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \textbf{here} \wedge \neg\exists y\, \textsc{Holding}(y) \supset \textbf{Res}(\textsc{pickup}, \textsc{Holding}(x)))$$

Definition 4.1 $\textbf{here} \overset{\text{def}}{=} \textsc{pos}(\textbf{self})$

It says that if some object x is positioned where the agent currently is and he is not currently holding anything, then his doing the action PICKUP next will result in his holding x.[11] This means that under these conditions, it is both physically possible for him to do PICKUP, and his doing so necessarily results in his holding the object. In fact, we assume that all basic actions are always possible. The view adopted is that such actions characterize essentially internal events which may have various external effects depending on the circumstances.[12] We also assume that agents always know how to do basic actions, that is, know what primitive actions they denote. This is formalized as follows:

Assumption 4.2 (Basic actions are known)

$$\models \exists d\, \textbf{Know}(d = \theta^d), \text{ where } \theta^d \in \{\textsc{pickup}, \textsc{putdown}, \textsc{forward}, \textsc{right}, \textsc{left}, \textsc{sense}\}$$

We also make various frame assumptions for PICKUP (i.e., assumptions about what does not change as a result of the action). The following says that when the agent does PICKUP, the positions of all things must remain unchanged:

Assumption 4.3 (PICKUP does not affect POS)

$$\models \forall x \forall p(\textsc{pos}(x) = p \supset \textbf{AfterNec}(\textsc{pickup}, \textsc{pos}(x) = p).$$

[10]Assumptions are essentially non-logical axioms in a theory of a particular domain (they are not part of the logic proper). In reasoning within a theory, we only deal with semantic structures where the assumptions come out true. Note that due to this, assumptions not only hold at time **now**, but at all times, and it is common knowledge [18] that this is the case (i.e., everyone knows it, everyone knows that everyone knows it, and so on).

[11]Even though we are specifically talking about the agent and time of the context in the above, the attribution in fact applies to all agents and times, since it is assumed that the assertion is valid (i.e., satisfied at all indices), and it is a property of our logic that if $\models \varphi$, then $\models \forall a \forall t \textbf{By}(a, \textbf{At}(t, \varphi)))$. If several agents were involved, we might have to formalize the domain differently.

[12]For instance, assumption 4.1 only specifies what happens when PICKUP is done under the conditions stated; what its effects are in other circumstances is not addressed.

We also assume that PICKUP does not affect the orientation of anything and that unheld objects that are not where the agent is remain unheld after the action; these assumptions are specified analogously to the one above and we omit them here.

Now clearly, just having *de re* knowledge of some object is insufficient for being able to pick it up; something must be known about the object's position. Perhaps there are domains where as soon as an agent knows which object is involved, he would know how to get to it (or how to find out); in such a case, one might want to suppress all aspects of the process by which the agent determines the object's relative position and navigates to it, and thus one might develop a formalization where the formula in proposition 4.1 turns out to be valid. However, this is clearly not the usual case; agents often fail to know where objects are. Our formalization reflects this; it models the situation at a level of detail sufficient to account for what agents need to know about the position of objects they want to manipulate and how they manage to get to them. So the following proposition holds:

Proposition 4.1 $\not\models \exists x \mathbf{Know}(\textsc{Object}(x)) \supset \mathbf{Can}(\textsc{pickup}, \exists x\ \textsc{Holding}(x))$

Proof:
Consider a structure where $\mathsf{e}, \mathsf{g} \models \mathbf{Know}(\textsc{Object}(x))$ but $\mathsf{e}, \mathsf{g} \not\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}))$, i.e., the agent the agent knows of some object, but does not know that this object is **here**, in fact, does not know that any object is **here**. Let there be an index e' be such that $\langle \mathsf{e}, \mathsf{e}' \rangle \in \mathsf{K}$ and $\mathsf{e}', \mathsf{g} \models \mathbf{Does}(\textsc{pickup}, t) \wedge \neg \mathbf{At}(t, \exists x\ \textsc{Holding}(x))$. This is consistent with assumptions 4.1 and 4.2. It is easy to see that the above implies that $\mathsf{e}, \mathsf{g} \not\models \mathbf{Know}(\mathbf{AfterNec}(\textsc{pickup}, \exists x\ \textsc{Holding}(x)))$ (using the fact that $\approx_{\mathsf{t}}$ must be reflexive), i.e., the agent does not know that after he does PICKUP, he must be holding something. So the agent is not able to achieve the goal by doing PICKUP and the structure falsifies the formula. ■

In a discussion of the robot action of "putting a block on another block", Moore [41] recognizes that knowing what blocks are involved may not be enough and suggests that the action be defined in terms of lower-level actions involving arm motions to the objects' positions, grasping, and ungrasping. Now, knowledge of an object's absolute position is not sufficient for being able to act upon it (and nor is it necessary). One may not know what one's absolute position and orientation is and therefore may not be able to deduce where the object is relative to oneself. Our formalization reflects this fact. For instance, one can prove the following proposition with respect to the simple situation discussed earlier. It says that even if the agent is currently at some position p and knows that the absolute position of some object is p and that he is not holding anything, he still might not be able to achieve the goal of holding some object by doing the action PICKUP. The reason for this is simply that he may not know that he is at p.

Proposition 4.2

$$\begin{aligned}&\not\models \exists p(\mathbf{here} = p \wedge \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = p) \wedge \neg \exists y\ \textsc{Holding}(y)))\\ &\quad \supset \mathbf{Can}(\textsc{pickup}, \exists x\ \textsc{Holding}(x))\end{aligned}$$

The proof is similar to that of the previous proposition; it appears in [29].

On the other hand, we can also prove that if the agent knows that some object is *where he currently is* and that he is not holding anything, then he must be able to achieve the goal of holding some object by doing PICKUP:

Proposition 4.3

$$\begin{aligned}&\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y))\\ &\quad \supset \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x))\end{aligned}$$

Proof:
Suppose that the antecedent holds at e and g. By assumption 4.2, we have that $\mathsf{e},\mathsf{g} \models \exists d\, \mathbf{Know}(d = \textsc{pickup})$, i.e., the agent knows what the action is. By quantifier reasoning, assumption 4.1 implies that

$$\begin{aligned}&\models \exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y) \supset\\ &\quad \mathbf{PhyPoss}(\textsc{pickup}) \wedge \exists x\, \mathbf{AfterNec}(\textsc{pickup}, \textsc{Holding}(x))\end{aligned}$$

So by propositions 3.3 and 3.2 and the supposition, it must be the case that $\mathsf{e},\mathsf{g} \models \mathbf{Know}(\mathbf{PhyPoss}(\textsc{pickup}))$, i.e., the agent knows that the action is physically possible, as well as that $\mathsf{e},\mathsf{g} \models \mathbf{Know}(\exists x \mathbf{AfterNec}(\textsc{pickup}, \textsc{Holding}(x))$. It is straightforward to produce a semantic proof that $\models \exists v \mathbf{AfterNec}(\theta^d, \varphi) \supset \mathbf{AfterNec}(\theta^d, \exists v \varphi)$ (provided that v does not occur in θ^d). Thus by propositions 3.3 and 3.2, it must be the case that $\mathsf{e},\mathsf{g} \models \mathbf{Know}(\mathbf{AfterNec}(\textsc{pickup}, \exists x \textsc{Holding}(x)))$, i.e., the agent knows after doing the action, the goal must hold. Thus by the definition, $\mathsf{e},\mathsf{g} \models \mathbf{Can}(\textsc{pickup}, \exists x \textsc{Holding}(x))$.
■

The agent can be totally ignorant of what his (and the object's) absolute position is and still be able to achieve the goal.

Note that proposition 4.3 makes no requirement that the object that the agent ends up holding be the same as the one that was at his position before the action. This may appear too weak and an easy fix would involve assuming that the agent knows *which* object is involved. But it is possible to strengthen the above proposition without requiring such *de re* knowledge. For example, the following proposition captures the fact that the agent knows that after the action, he would be holding some object that was where he was before doing the action. Specifically, it says that if the agent knows that some object is currently at his position and that he is not currently holding anything, then he can by doing action PICKUP achieve the goal of holding some object that was at his own position before the PICKUP he has just done.

Proposition 4.4

$$\begin{aligned}&\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here}) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset\\ &\quad \mathbf{Can}(\textsc{pickup}, \exists x(\textsc{Holding}(x) \wedge \mathbf{DoneWhen}(\textsc{pickup}, \textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here})))\end{aligned}$$

The proof is similar to that of proposition 4.3; it appears in [29]. This result can be strengthened further to require uniqueness. But it should be clear that identifying the objects involved in the initial and goal situations, without requiring that it be known what objects they are, is not a trivial matter.

Before moving on, let's examine another variant of this situation. By the necessitation rule for **By** and universal generalization, an immediate consequence of proposition 4.3 is that any agent who knows that there is an object where he himself is and that he is not holding anything must be able to achieve the goal of holding something by doing PICKUP, that is, $\models \forall a \mathbf{By}(a, \varphi_p)$, where φ_p is the formula of proposition 4.3. However, if instead of this, an agent a merely knows that there is an object where a is, it no longer follows that he is able to achieve the goal.

Proposition 4.5

$$\begin{aligned}\not\models \forall a \mathbf{By}(a, &\mathbf{Know}(\exists x(\textsc{Object}(x) \land \textsc{pos}(x) = \textsc{pos}(a)) \land \neg\exists y\, \textsc{Holding}(y)) \\ &\supset \mathbf{Can}(\textsc{pickup}, \exists x\, \textsc{Holding}(x)))\end{aligned}$$

The reason why this is not valid is simply that a may not know that he is a. This shows that knowing of oneself (*de re*) that if one does the action, the goal will necessarily hold afterwards, as Moore's formalization of ability requires, is not sufficient for ability. One can similarly show that such *de re* knowledge is not necessary either (in some models of proposition 4.3, the agent does not have such knowledge).

4.3 Going to a Relative Position

Let us now look at navigation. Since there are no obstacles in our robot's world, it seems that given any relative position, the robot should be able to go there. Let us show formally that this is the case. The effects of action FORWARD are specified as follows:

Assumption 4.4 (Effects of FORWARD**)**

$$\models \forall p \forall o(\mathbf{here} = p \land \mathbf{selfori} = o \supset \mathbf{Res}(\textsc{forward}, \mathbf{here} = p + \langle 1, 0\rangle \times \textsc{rot}(o)))$$

Definition 4.2 $\mathbf{selfori} \overset{\text{def}}{=} \textsc{ori}(\mathbf{self})$

Definition 4.3 $\textsc{rot}(\theta^i_o) \overset{\text{def}}{=} \left\langle \begin{array}{cc} \cos\theta^i_o & \sin\theta^i_o \\ -\sin\theta^i_o & \cos\theta^i_o \end{array} \right\rangle$

Assumption 4.5 $\models \forall a(0 \leq \textsc{ori}(a) < 2\pi)$

Assumption 4.6 $\models \forall x(\textsc{Holding}(x) \supset \textsc{pos}(x) = \mathbf{here})$

Assumption 4.4 says that as a result of doing FORWARD, the agent moves one square further along the direction he is facing; **selfori** represents the orientation of the agent with respect to the absolute frame of reference and $\textsc{rot}(\theta^i_o)$ is the rotation matrix associated with angle

θ^i_o. Assumption 4.6 says that objects held by the agent are where he is. We also need the following frame assumptions: that the agent's doing FORWARD, must leave his orientation unchanged (a formal version of this appears in appendix A.1), that it must leave the position of objects that are not held by him unchanged; and that it has no effect on whether or not an object is being held.

Notice that we use expressions from the language of arithmetic and analytic geometry in formalizing this domain. We make the following assumptions regarding these forms:

Assumption 4.7
All the constants and function symbols from arithmetic and analytic geometry used are rigid (i.e., have the same denotation at every index).

Assumption 4.8 All facts from arithmetic and analytic geometry are valid.[13]

A consequence of assumption 4.7 is that the restrictions on specialization do not apply for arithmetic or analytic geometry expressions:

Proposition 4.6

$\models \forall v\varphi \supset \varphi\{v \mapsto \theta\}$, provided that θ is free for v in φ and all constants and function symbols θ contains are arithmetic or from analytic geometry

Finally, let us add a few more definitions:

Definition 4.4 $\textsc{rposToApos}(\theta^i_p) \stackrel{\text{def}}{=} (\textbf{here} + \theta^i_p \times \textsc{rot}(\textbf{selfori}))$

Definition 4.5 $\textsc{roriToAori}(\theta^i_o) \stackrel{\text{def}}{=} \textsc{mod}_{2\pi}(\textbf{selfori} + \theta^i_o)$

Definition 4.6

$$\begin{aligned}
&\textsc{MovedToBy}(\theta^i_p, \theta^i_o, \delta) \stackrel{\text{def}}{=} \\
&\quad \exists v^i_p \exists v^i_o(\textbf{DoneWhen}(\delta, v^i_p = \textsc{rposToApos}(\theta^i_p) \wedge v^i_o = \textsc{roriToAori}(\theta^i_o)) \\
&\qquad\qquad \wedge\ \textbf{here} = v^i_p \wedge \textbf{selfori} = v^i_o)
\end{aligned}$$

where v^i_p and v^i_o do not occur in θ^i_p, θ^i_o, or δ

$\textsc{rposToApos}$ is a function that converts a relative position into an absolute one; similarly, $\textsc{roriToAori}$ converts a relative orientation to an absolute one. $\textsc{MovedToBy}(\theta^i_p, \theta^i_o, \delta)$ means that the agent has just done action δ and that by doing so, he has moved to the square that was at relative position θ^i_p (before the action) and has turned towards the direction that was relative orientation θ^i_o (before the action).

Now, let us discuss what these assumptions imply with respect to the ability of the robot to get to a given relative position. We start with a simple case: that the robot is able to go

[13]This is essentially a placeholder for an axiomatization of arithmetic and analytic geometry; there should be no problem coming up with one that gives us all the facts we need even though it would necessarily be incomplete.

to the square directly in front of him (i.e., to relative position $\langle 1,0\rangle$) by doing FORWARD. In fact, the following proposition says something slightly stronger: that by doing FORWARD, one is able to achieve the goal of being on the square that was at relative position $\langle 1,0\rangle$ before the action and of having one's orientation remain unchanged.

Proposition 4.7 $\models \textbf{Can}(\text{FORWARD}, \text{MovedToBy}(\langle 1,0\rangle, 0, \text{FORWARD}))$

A proof of this proposition is in appendix A.1.

This result can then be extended to deal with squares arbitrarily far away along the row that the agent is facing (i.e., relative positions $\langle \mathsf{n},0\rangle$, where $\mathsf{n} \in \mathbb{N}$). We can show that by doing FORWARD n times in sequence, one is able to achieve the goal of being on the square that was at relative position $\langle \mathsf{n},0\rangle$ before the action and of having one's orientation remain unchanged:

Proposition 4.8

$$\text{For all } \mathsf{n} \in \mathbb{N},\ \models \textbf{Can}(\text{FORWARD}^{\mathsf{n}}, \text{MovedToBy}(\langle \mathsf{n},0\rangle, 0, \text{FORWARD}^{\mathsf{n}}))$$

The proof is in appendix A.2.

Now, let us look at the general case. The action of going to a relative position is defined below. The definition goes as follows: to go to relative position $\langle \mathsf{n},0\rangle$, where $\mathsf{n} \in \mathbb{N}$, one simply goes forward n times; to go to relative position $\langle \mathsf{m},0\rangle$, where m is a negative integer, that is, to a position behind oneself, one turns left twice (i.e., 180°), then goes to relative position $\langle -\mathsf{m},0\rangle$, and then one turns left twice again, so as to return to one's original orientation; finally, to go to an arbitrary relative position $\langle \mathsf{n},\mathsf{m}\rangle$, one first goes to $\langle \mathsf{n},0\rangle$, that is, to the right position on the x axis, then one turns left 90°, then one goes to $\langle \mathsf{m},0\rangle$, and then finally, one turns left three times (i.e., 270°) to return to the original orientation.

Definition 4.7

$\text{GORPOS}(\langle \mathsf{n},0\rangle) \overset{\text{def}}{=} \text{FORWARD}^{\mathsf{n}}$, where $\mathsf{n} \in \mathbb{N}$

$\text{GORPOS}(\langle \mathsf{m},0\rangle) \overset{\text{def}}{=} \text{LEFT}^2; \text{GORPOS}(\langle -\mathsf{m},0\rangle); \text{LEFT}^2$, where $\mathsf{m} \in \mathbb{Z}, \mathsf{m} < 0$

$\text{GORPOS}(\langle \mathsf{n},\mathsf{m}\rangle) \overset{\text{def}}{=} \text{GORPOS}(\langle \mathsf{n},0\rangle); \text{LEFT}; \text{GORPOS}(\langle \mathsf{m},0\rangle); \text{LEFT}^3$, where $\mathsf{n},\mathsf{m} \in \mathbb{Z}, \mathsf{m} \neq 0$

We also need provide a specification for the action of turning left. Its effects are formalized as follows:

Assumption 4.9 (Effects of LEFT)

$$\models \forall o(\textbf{selfori} = o \supset \textbf{Res}(\text{LEFT}, \textbf{selfori} = \text{MOD}_{2\pi}(o + \pi/2)))$$

This says that as a result of doing LEFT, the agent's orientation is rotated by 90°. We also make the frame assumptions that doing LEFT does not affect the position of anything nor whether any object is being held. Given this, it is straightforward to prove ability results for LEFT similar to the ones previously obtained for FORWARD. Let us only state our main result:

Proposition 4.9

$$\text{For all } \mathsf{n} \in \mathbb{N},\ \models \mathbf{Can}(\textsc{left}^{\mathsf{n}}, \textsc{MovedToBy}(\langle 0,0\rangle, \textsc{mod}_{2\pi}(\mathsf{n}\pi/2), \textsc{left}^{\mathsf{n}}))$$

This says that by doing LEFT n times in sequence, the agent is able to achieve the goal of having his orientation rotated by n times 90° to the left of what it was before the action and of having his position remain unchanged.

Then more generally, one can prove the following proposition about GORPOS; it says that one can, by doing $\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle)$, achieve the goal of being on the square that was at relative position $\langle \mathsf{n}, \mathsf{m}\rangle$ before the action and of having one's orientation remain unchanged:

Proposition 4.10

$$\begin{aligned}&\text{For all } \mathsf{n}, \mathsf{m} \in \mathbb{Z},\\ &\models \mathbf{Can}(\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle), \textsc{MovedToBy}(\langle \mathsf{n}, \mathsf{m}\rangle, 0, \textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle)))\end{aligned}$$

A proof of this proposition is in appendix A.3.

Let us also point out that our formalization captures the fact that an agent need not be able to go to a given absolute position, for example, in a situation where he does not know where he is. The following proposition formalizes a simple instance of this; it states that the agent need not be able to go to absolute position p by doing FORWARD, even if p happens to be the position of the square directly in front of him.

Proposition 4.11 $\not\models \forall p(p = \textsc{rposToApos}(\langle 1, 0\rangle) \supset \mathbf{Can}(\textsc{forward}, \mathbf{here} = p))$

A common reason for going somewhere is that one wants to get hold of or manipulate something that is there. It is straightforward to extend the results obtained so far and show that knowing the relative position of an object is sufficient for being able to go and manipulate it. Let $\textsc{rpos}(\theta^i)$ stand for the position of θ^i relative to **self**:

Definition 4.8 $\textsc{rpos}(\theta^i) \stackrel{\text{def}}{=} ((\textsc{pos}(\theta^i) - \mathbf{here}) \times \textsc{rot}(-\mathbf{selfori}))$

In [29], we prove the following result:

Proposition 4.12

$$\begin{aligned}&\text{For all } \mathsf{n} \in \mathbb{N},\\ &\models \mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle \mathsf{n}, 0\rangle) \wedge \neg\exists y\, \textsc{Holding}(y)) \supset\\ &\quad \mathbf{Can}((\textsc{forward}^{\mathsf{n}}; \textsc{pickup}), \exists x\, \textsc{Holding}(x))\end{aligned}$$

This says that if the robot knows that there is an object at position $\langle \mathsf{n}, 0\rangle$ relative to himself, that is, on the n'th square directly in front of him, and knows that he is not holding anything, then he is able to achieve the goal of holding some object by doing FORWARD n times followed by PICKUP. Note that this result requires the additional frame assumption that whether an entity is an object is not affected by any action. We omit details as no new techniques are required. It should be easy to generalize this result and show that by doing $\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m}\rangle)$ followed by PICKUP, the agent can come to hold something, provided that he knows that there is an object at relative position $\langle \mathsf{n}, \mathsf{m}\rangle$ and that he is not holding anything.

4.4 Perception

We will come back to this issue of what one *must* know in order to be able to go and manipulate an object, but now let's have a look at perception. As observed earlier, it too yields indexical knowledge.[14] In our domain, the action SENSE constitutes a limited form of perception. We formalize the effects of SENSE as follows:

Assumption 4.10

$$\models \forall s \mathbf{Res}(\textsc{sense}, \mathbf{Kwhether}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\textsc{Holding}(x) \wedge \textsc{OfShape}(x, s))))$$

This assumption says that doing SENSE results in the agent knowing whether an unheld object of a given shape is present at his current position (we are assuming that there can only be a single object resting on a square and that the robot's sensors are focussed on this object and not on any object that he might be holding). From this assumption and the fact that basic actions are assumed to be known, it follows trivially that by doing SENSE, the agent can find out whether there is an unheld object where he is and, if there is one, what its shape is.

We also make the following frame assumptions: that the agent's doing SENSE leaves his orientation and the positions of everything unchanged; that it does not change whether an object is held or not; and finally that no action affects the shape of objects.

4.5 Ability to Go Where an Object Is

Let's now go back to the issue of what one must know in order to be able to go and act upon an object. We said that knowing the relative position of the object was sufficient for this. But in real life, agents rarely know exactly what the relative positions of objects are. More typically, they know roughly where objects are and scan the general area until they find the object. The following proposition formalizes an instance of this; it says that by sensing and then scanning forward, up to k squares, until he senses that an unheld object is present, the robot can achieve the goal of either knowing that there is an unheld object where he is, or knowing that there are no unheld objects on the k squares behind him (including the one he is currently on). The $\textsc{scan}_{\mathsf{k}}(\varphi)$ action involves repetitively moving forward and sensing (up to k times) until φ becomes true.

Definition 4.9 $\textsc{scan}_{\mathsf{k}}(\varphi) \overset{\text{def}}{=} \mathbf{while}_{\mathsf{k}}(\neg\varphi, \textsc{forward}; \textsc{sense})$

Proposition 4.13

For all $\mathsf{k} \in \mathbb{N}$,

$$\begin{aligned}\models \mathbf{Can}(&(\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\textsc{Holding}(x)))),\\ &\mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\textsc{Holding}(x))) \vee\\ &\mathbf{Know}(\neg\exists n(-\mathsf{k} \leq n \leq 0 \wedge \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle n, 0\rangle \wedge \neg\textsc{Holding}(x)))))\end{aligned}$$

[14] Davis [10, 12, 11] has done interesting work on the topic of reasoning about knowledge and perception; however, he does not address the fact that the knowledge obtained from perception is indexical knowledge.

Appendix A.4 contains a proof of this proposition.

So it is quite clear that ability to act upon an object does not require knowing its relative position. But then what is required? It seems that the best we can say is that the agent must *know of some procedure* that will take him to where the object is.

This creates problems in formalizing the knowledge prerequisites of certain high-level parametrized actions, for example, the action of "going to the position of an object θ" $\textsc{goWhere}(\theta)$. It would be inappropriate to treat this action as a primitive because we want to model how knowledge enables action at a more detailed level. The other way of dealing with such an action within our (and Moore's) framework involves defining it in terms of lower-level actions that are parametrized with the information that must actually be known in order to be able to do the high-level action. This allows knowledge prerequisites to be enforced by the requirement that one know which primitive action to do next. But for actions like $\textsc{goWhere}(\theta)$, it is not clear how this can be put into practice.

However, notice that $\textsc{goWhere}(\theta)$ is a strange kind of action, in that it appears to refer to anything that would achieve the goal that the agent be where θ is; it behaves as much as a *goal* as like an action. Perhaps we should rule out the introduction of such actions, but instead provide an action-less version of the **Can** operator: $\mathbf{CanAch}(\varphi)$ would mean that **self** is somehow able to achieve φ. Then, we may use $\mathbf{CanAch}(\textsc{pos}(\theta) = \mathbf{here} \wedge \varphi)$ instead of something like $\mathbf{Can}(\textsc{goWhere}(\theta), \varphi)$.[15] A coarse "syntactic" way of formalizing **CanAch** goes as follows: $\mathsf{e}, \mathsf{g} \models \mathbf{CanAch}(\varphi)$ iff there exists an action expression δ such that $\mathsf{e}, \mathsf{g} \models \mathbf{Can}(\delta, \varphi)$. A more general and robust approach has been proposed by Nunes and Levesque [46].

4.6 Map Navigation

A general account of ability must be based on a formalism that handles both indexical knowledge and objective knowledge, as well as knowledge that relates the agent's perspective to the objective frame of reference, what we might call *orienting knowledge*. To see why this is the case, let us look at one last area of robotics, that of navigation with the help of a map. Maps (of the usual kind) contain objective information. To use this information, say to figure out how to get to a destination, an agent must first orient himself — find out where he is on the map, that is, what his absolute position is, and find out what his absolute orientation is. If he does not already have this information, he might try to obtain it by searching his environment for landmarks represented on the map.

Our simple domain provides instances of this if we treat objects of various shapes as landmarks. Consider a variant of the scanning example of the previous section, where the robot knows what the absolute position of the unique object of shape s is (from having examined the map); then by scanning forward, the robot should be able get into a state where he either knows where he is, or knows that the object is not in the region scanned. The following proposition shows that this is essentially correct. It says that if the robot

[15]This assumes that it is known that θ refers to the same entity before and after the action; the assumption can be dispensed with by referring to the denotation of θ prior to the action as illustrated in section 4.2.

knows that the unique object with shape s is at absolute position p and that it is not held, then by sensing and then scanning forward, up to k squares, until he senses that an unheld object with shape s is present, he can achieve the goal of either knowing that he is at absolute position p (i.e., knowing where he is), or knowing that the unheld object of shape s is on none of the k squares behind him (including the one he is currently on).

Proposition 4.14

For all $\mathsf{k} \in \mathbb{N}$,

$$\begin{aligned}
&\models \forall p \forall s(\mathbf{Know}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = p \wedge \neg\textsc{Holding}(x) \wedge \\
&\qquad\qquad \forall y(\textsc{OfShape}(y,s) \equiv y = x))) \supset \\
&\quad \mathbf{Can}((\textsc{sense};\textsc{scan}_{\mathsf{k}}(\exists x(\textsc{Object}(x) \wedge \textsc{pos}(x) = \mathbf{here} \wedge \neg\textsc{Holding}(x) \wedge \textsc{OfShape}(x,s)))), \\
&\qquad \mathbf{Know}(\mathbf{here} = p) \vee \\
&\qquad \mathbf{Know}(\neg\exists n(-\mathsf{k} \leq n \leq 0 \wedge \exists x(\textsc{Object}(x) \wedge \textsc{rpos}(x) = \langle n, 0\rangle \wedge \neg\textsc{Holding}(x) \\
&\qquad\qquad\qquad \wedge\ \textsc{OfShape}(x,s))))))
\end{aligned}$$

The proof is omitted as it is similar to that of proposition 4.13; a compete proof of a closely related result appears in [29].

Similarly, an agent might find out what its absolute orientation is by searching for a pair of landmarks whose orientation relative to one another is indicated on the map, or perhaps by searching for a single landmark whose faces have distinctive shapes that are represented on the map. Once the agent has found out where he is and what his orientation is, he will know which relative position corresponds to any absolute position; he can then use the information in the map to figure out how to go where he wants.[16] Orienting knowledge, that is, knowledge about how indexical notions are related to objective notions, allows objective knowledge to be mapped into indexical knowledge and vice versa. It is a key feature of our formalism that it allows all these kinds of knowledge to be represented.

5 Other Applications

The notion of indexical knowledge and the important role it plays in action are most easily understood by looking at robotics examples as we did in the previous section. These involve actions taking place in physical space; the agent has knowledge that is relative to his point of view in physical space. This might suggest that the notion is bound to that of physical space. But this is not the case; the notion is really rather abstract. It is useful as long as the domain involves agents that operate in some kind of space, from some kind of point of view into that space.

In [29], two non-robotics domains that involve very abstract notions of space are examined. The first involves an agent making a phone call. The phone system is naturally viewed as a kind of space structured into various kinds of regions and sub-regions. An agent can be quite ignorant of what characterizes his absolute position in that space, things like area code, country code, etc. This domain was formalized and it was proven that an agent

[16] Many of these characteristics of the map navigation problem were pointed out by Israel [22].

is able to establish a connection to some phone if he knows what its number is and either knows that it is in his own area, or knows that it is not and knows what the phone's area code is (international and same-area long distance calls were ignored).

The other domain examined is that of data structure search and manipulation. At first, it might seem strange to look at this in terms of an agent located in some kind of space and having knowledge that is relative to his point of view into that space, yet this seems very much the outlook taken by many algorithms; one talks about following pointers and searching graphs. An example involving the heapsort algorithm [4] was partly formalized. We sketched a proof of the fact that an agent is able to heapify a tree consisting of a new node installed as root of two existing heaps by sifting the new node down the tree until its key is at least as large as that if its sons.[17] Note that the "space" involved in this example has a very different topology from that of physical space.

There is as much of a need for keeping track of one's place in time as of one's place in space. In the spatial examples of section 4, we described cases where the agent is required to know his absolute position, other cases where he need only know where something is relative to himself, that is, know its relative position, and still other cases where neither kind of knowledge is needed — the agent needs only know a navigation procedure that gets him where he wants. The same kind of distinctions apply to situations where temporal knowledge is required for ability. Depending on the case, the agent might need to:

1. *know what time it is,* e.g., an agent that needs to lock up a room at a specific time, say 5 p.m., and does so by checking a clock until the time comes,

2. *know how much time some process needs to go on,* e.g., an agent who soft-boils an egg (i.e., cooks it without having the yolk solidify) using a timer,

3. *monitor the environment for a condition indicating that a process has gone on long enough,* e.g., an agent who wants to fry a fish filet without overcooking it and who achieves this by frying it until its flesh is no longer translucent.

In [29], a formalization of the first two examples is sketched. The third example does not require time to be explicitly represented; it can be formalized much like the scanning example of section 4.5. In [32], we also formalize an example of a common temporal reasoning/planning problem that can only be handled in a formalism that includes both indexical and non-indexical concepts and supports reasoning using both. The example involves an agent that does not initially know what time it is; he must keep track of time in relative terms (using a timer), but later convert this indexical knowledge into absolute knowledge (by finding out what time it is) for communication to another agent.[18] It was found that the

[17]A heap is a binary tree that satisfies the "heap property", that is, where the key stored at any node is greater or equal to the ones stored at its left and right sons (if they exist).

[18]Specifically, the agent has to set a turkey to roast in the oven and leave a message saying when the turkey started roasting, so that someone else can take it out when it is ready. But the agent (who sets the turkey to roast) does not initially know what time it is and only has access to the one-hour timer on the stove and a radio station that announces the time at least every 30 minutes. The plan we consider involves

framework proposed provides the necessary tools for producing a satisfactory formalization in every one of these cases.

6 The Need for Distinguishing Between Indexical and Objective Knowledge

To someone accustomed to the objective point of view of science or mathematics, indexicality (context-sensitivity) may appear as little more than an artifact of natural language. One may thus claim that while using indexical descriptions is often convenient, in practice, indexical knowledge can always be understood objectively. One reason for wishing that this claim were true has to do with the fact that the semantic content of indexical representations depends on the context, so if the context changes you may have to adjust the representations to keep the semantic content unchanged. For instance, if an agent's knowledge base describes some facts as holding 'now', then at the next time step, it should describe these facts as holding 'one time step before now'.[19] Haas [17] points out that the cost of adjusting a knowledge base that contains indexical time references for the passage of time would be high, if implemented in the obvious way. He proposes that a robot use its internal clock to eliminate all occurrences of 'now' in its representations.

Let us discuss the claim that indexical knowledge can be reduced to objective knowledge. In our semantics for knowledge, indexical terms and formulas are treated as relative to an agent and a time; for example, knowing that something is **here** amounts to knowing that something is at one's position at the current time. Given this, it is clear that if one knows who one is and knows what time it is (remember that we are taking *de re* knowledge to require knowing a standard name), then anything that one knows in an indexical way is also known in an objective way.[20] But is it reasonable to assume that an agent always knows who he is and what time it is?

Let's consider the temporal part of this question (see [32] for a more detailed discussion). First, humans do not have internal clocks that they can use in the way a robot can, and they do not always know what time it is. A system that is interacting with humans will need to model this (e.g. to remind a user that his meeting is just starting). Even if we limit ourselves to simple robotics contexts, it seems unlikely that the internal clocks of robots could be guaranteed to be accurate. In such cases, Haas's scheme leads to indexical information being misrepresented. Moreover, Haas's robot cannot even represent the fact that his internal clock is incorrect; it could not monitor for this and plan to get its clock reset when appropriate. Also, for very simple (e.g. insect-like) robots, the cost of fitting

putting the turkey in the oven, setting the timer to one hour, then listening to the radio until the time is announced while keeping track of the roasting time with the timer, and finally calculating the time the turkey started roasting, and leaving a message to that effect.

[19] Subramanian and Woodfill [53] prove that such a transformation is truth-preserving within their indexical situation calculus framework.

[20] E.g., if at 10 a.m. Dec. 8, 1991, Rob the robot knows that there is currently a pop can one meter in front of him, and also knows who he is and what time it is, then he must knows of Rob and of 10 a.m. Dec. 8, 1991 that there is a pop can one meter in front of that person at that time (*de re*).

them with internal clocks and setting them may be too high. Finally, as Haas recognizes, it is not at all clear that the cost of updating a temporal knowledge base that contains indexicals need be prohibitive; for example, if all occurrences of 'now' are replaced by a new constant and the fact that this new constant is equal to 'now' is added, then only this single assertion need be updated as time passes.

Work on reactive agent architectures supplies other reasons for wanting a formalism that can represent indexical knowledge. As pointed out by Agre and Chapman [2], the world can change in unexpected ways and reasoning about change can be very costly; in some cases it is better to rely on perception to get fresh information at every time step rather than try to update a representation of the world; in such cases, the problem of updating indexical representations does not arise. And as Rosenschein and Kaelbling [49] have shown, it is legitimate to ascribe knowledge to agents even when they have no explicit representation of this knowledge. In such cases, one needs a formalism that distinguishes between indexical and objective knowledge just to accurately model the agent's thinking. The output of the agent's perception module says nothing about time, and even if the agent has a correct internal clock, he may have no need to time-stamp his knowledge. We want a formalism that makes the distinctions required to model this.

Let us briefly discuss the other half of the above question, that is, whether agents need always know who they are (know a standard name for themselves). It is tempting to dismiss the usual amnesia examples as mere philosophical curiosities. But if we think of very simple agents, say insects; it would not seem all that unusual to have them not know who they are; in fact, one is hard pressed to come up with good reasons for them to need to know who they are. One can also imagine mass produced computers or robots that do not have their name etched in memory at the factory or even entered at boot-time. One might also look at processes running in a computer as agents. Grove [14] and Grove and Halpern [15] describe cases in the area of distributed systems and communication protocols where one does not want to assume that agents know who they are. One of their examples involves a set of processes that are running a leader-election protocol (to select one of them to play some special role later on); the processors are anonymous, they do not have any unique identifier as part of their state and they are all running the same program; they do not know who they are.

7 Conclusion

Agents act upon and perceive the world from a particular perspective. It is important to recognize this relativity to perspective if one is not to be overly demanding in specifying what they need to know in order to be able to achieve goals through action. In many cases (especially simple interactions with their physical environment) they need not know much about their objective situation; what they need is *indexical knowledge*, knowledge about how they are related to things in their environment or to events in their history. And perception yields just the kind of indexical knowledge that is needed.

Previous formal accounts of the ability of agents to achieve goals by doing actions, such as that of Moore [40, 41] and Morgenstern [42, 43], have ignored this, and thus end up imposing knowledge requirements that are neither necessary nor sufficient for ability; they fail to properly specify the knowledge prerequisites and effects of actions. In this work, we have developed a formal solution to the problem of how one should model the indexical knowledge prerequisites and effect of action. We have also shown how the theory can be used to formalize several common types of interaction between a robot and its environment.

7.1 Contributions

Let us review the ways in which our theory improves over previous accounts of the relationship between knowledge and action. Our account of knowledge formally captures the distinction between indexical and objective knowledge: it allows an agent to know something in an indexical way without knowing it in any kind of objective way and vice-versa. Knowledge about how objective notions are related to indexical notions can be expressed, thus allowing objective knowledge to be mapped into indexical knowledge and vice-versa. Also, the account fully handles time, in particular, temporally indexical knowledge. Its simple modal syntax provides succinct ways of expressing most matters of interest, in particular, attributions of indexical knowledge. Its model-theoretic semantics is a simple and natural extension of standard possible-world semantic schemes for the logic of knowledge. It retains what is perhaps the most attractive feature of possible-world semantics: the correspondence between constraints on the accessibility relation and important properties of the notion formalized. On the other hand, it also inherits some limitations of the possible-world approach, in particular, the "logical omniscience" problem (i.e., agents are taken to know all the logical consequences of their knowledge, as well as all validities). Finally, our account includes a formalization of the requirement that knowledge be persistent that works well with both temporally indexical and temporally absolute knowledge.

The temporal subset of our logic is simple and very expressive. Both eternal and indexical temporal assertions can be made. Relations between indexically specified and objectively specified times can also be expressed. The fact that terms denoting times are included allows quantification over times into epistemic contexts, so one can make very weak ascriptions of temporal knowledge; for example, one can express the claim that John knows that Paul knows what time it is without John himself knowing it with the formula $\mathbf{Know}(\textsc{john}, \exists t \mathbf{Know}(\textsc{paul}, \mathbf{now} = t))$. The occurrence of concurrent actions, continuous processes, and actions involving several agents can be expressed (even though our account of ability does not deal with these cases). There is one significant limitation: one cannot express the occurrence of actions involving indefinite iteration (unbounded while-loops). More generally, our account of the logic is limited by the fact that we have not identified a full axiomatization (even though the set of properties identified in [33, 29] is an important step in that direction); in fact, we do not know whether the set of valid formulas is recursively enumerable.

Our formalization of ability improves over previous accounts in several ways. Firstly, it does not require an agent to know who he is in order to be able to achieve a goal by doing an action; all references to the agent within the knowledge required are indexical references.

For instance, in the simple action case we require the agent to know that it is physically possible for *himself* to do the action and that if *he himself* does the action, the goal will necessarily hold afterwards (as well as knowing what the action is). *De re* knowledge of oneself is neither necessary nor sufficient for the agent to be able to achieve the goal (a concrete example was given in section 4.2). Situations where an agent does not know who he is (in the sense of not knowing an objective standard name for himself) are perhaps unusual, but our theory could not claim to reflect a real understanding of indexicality if it did not deal with such cases.

Secondly, our account of ability is based on a very expressive temporal logic. We can thus handle prerequisites or effects of actions that involve knowledge of absolute times and knowing what time it is, as well as many cases of actions that refer to times (e.g., the action of "locking up at 5 p.m.", discussed in section 5). This would also make it easier to extend the account to handle more complex actions than are currently treated, for instance concurrent actions, actions involving several agents, and actions that refer to time in very general ways.

Finally, the logic on which the account of ability is based includes an adequate treatment of indexical knowledge in general; as the applications in sections 4 and 5 show, this allows a more accurate specification of the knowledge prerequisites and effects of actions.

On the other hand, our account of ability suffers from various limitations. The class of actions handled is quite restricted. Moreover, the notion modeled is extremely idealized: it requires that the action absolutely guarantee that the goal will be achieved — a high probability of success is not enough.

We have shown how the framework can be used to model various common types of interaction between a robot and its environment. These examples show how actions can be formalized so as to avoid making excessive requirements upon the knowledge of agents — so that only indexical knowledge, as opposed to objective knowledge, is required when that is sufficient. Our formalization accounts for the facts that perception yields indexical knowledge, and that ability to act upon an object does not require knowing which object is involved or what its absolute position is. It was also shown how indexical knowledge and objective knowledge can be related in the framework, to deal with the use of maps for navigation. Many representational issues that have general relevance to the formalization of actions with indexical knowledge prerequisites or effects were discussed. Applications of the theory in other domains, in particular, ones that involve temporal knowledge, were also briefly discussed. These applications provide evidence to that effect that the distinction between indexical and objective knowledge supported by our framework has substantial practical value and that it cannot be done justice within existing accounts of ability.

7.2 Directions for Future Research

A major difficulty in producing a usable framework for reasoning about action is the frame problem, that is, providing a way of specifying actions that does not require enumerating all properties that are not affected by them. Recently, Reiter [48] has proposed a solution to the frame problem within the situation calculus and Scherl and Levesque [50] have extended this solution to deal with knowledge and knowledge-producing actions. The approach allows a form of regression to be used to reduce reasoning about what facts hold in a situation to reasoning about what facts held in an initial situation. We are currently reformulating our framework into an extended version of the situation calculus so as to incorporate this approach to the frame problem [51].

As mentioned earlier, our formalization of ability has many limitations. It should be extended to handle more complex types of actions, such as those involving indefinite iteration, nondeterminism, and concurrency, as well as plans involving multiple agents. Morgenstern [42, 43] as well as Nunes and Levesque [46] handle some of these cases, but their accounts do not deal with indexicality. Situations involving interacting agents are especially interesting from the indexicality point of view, since the difference in perspective between agents must be accounted for if they are to have indexical knowledge of the same facts. For example, imagine that we are cooperating in the dismantling of a motor, and are facing each other; suppose we need to jointly use a certain wrench in the next step; I might have to come to know that the wrench we are to use is the one on my left while you need to realize that it is the one on your right. However, multiple agents settings give rise to new difficulties for reasoning about change. One issue is whether agents know about all event occurrences that could affect their situation and if not, what assumptions they make about such events. Morgenstern [44] has proposed an approach to deal with this. Another issue connected to indexicality is the question of how agents refer to each other. In many cases, agents know of each other only under some indexical description (e.g., "the person in front of me now"); it would be inappropriate to assume that agents always know who all the agents involved are. The formalization proposed in [51] deals with this.

It would also be desirable to untangle various aspects of the notion of ability — distinguish between "being able to achieve a goal" and "knowing how to execute an action", between cases where the agent is aware of his ability and cases where he is not, between cases where the agent acts as a dumb interpreter of a program and cases where he does some optimization, etc. We are currently working on a formalization that deals with these distinctions, handles indefinite iteration and non-determinism, and is compatible with the approach to the frame problem mentioned earlier [34]. As well, the formalization of section 3.6 requires that it be known that performing the action absolutely guarantees that the goal will be achieved. Yet, in most real situations, agents cannot hope to attain such a degree of certainty. It would be desirable to come up with an account that can cope with this. Both default and probabilistic approaches should looked at.

It would also be desirable to have a mathematically satisfactory axiomatization of the logic — one that is complete, if this is achievable. This might require a re-examination of some aspects of the logic.

Another issue that should be investigated is the interaction between indexicality and hierarchical plans. It seems that the top level actions in such a plan would usually be specified in a relatively objective way (e.g., go to City Hall), while the lower level actions would be indexically specified (e.g., turn right at the next light). How do the different levels in such a plan relate to each other? Route planning should be a good test domain.

Indexicality is but one aspect of situatedness. One should look at how other aspects can be modeled. For instance, Barwise [5] talks about how "situated inference" is often relative to background conditions; as an example, he describes how one might infer that an object will fall from the fact that it has been released in mid-air, an inference that is relative to the presence of gravity. In many cases, such background conditions play a role in ensuring that an agent's doing some action will achieve his goal, but the agent is totally unaware of it, or he simply ignores it because these conditions normally hold in his usual environment. It should be possible to extend our account to model this notion of "ability relative to background conditions". More challenging would be developing a propositional attitude model of the kind of opportunistic acting/planning that Agre and Chapman describe in [3], but not obviously beyond current techniques.

The theory holds great potential for applications in the modeling of communication (both natural-language and that involving artificial agents exchanging messages in designed languages), especially in conjunction with the extensions mentioned earlier. We have seen that the knowledge required for many types of action is indexical. Thus while many uses of indexical expressions in language are only communicative shortcuts, ways of succinctly referring to the relevant entities, it should not be surprising that there are also cases where the information that needs to be communicated is intrinsically indexical. For example, I can help you get to my place by telling you where it is *relative to your current position*. To model this, one needs a formal account of communication that relates the context sensitivity of language to that of mental states and action.[21] In [30], a preliminary version of such an account is developed, using our theory of indexical knowledge and action as a foundation. Some recent work [55] has focussed on providing agents that interact with the world with the ability to understand natural language instructions; this would be an ideal setting for exploring concrete applications of such an account.

Last but not least, the theory would appear to hold much promise for applications in the design of reactive agents. Rosenschein and Kaelbling [49, 24] use a logical formalism as a design notation and as a specification language in their robot design framework. Since indexical knowledge appears to be centrally involved in the production of reactive behavior [2, 53], it seems that elements of our logic could prove useful in these roles. It might even be useful as a representation language for a planner or sophisticated executor. This kind of application might also yield back some insights into indexicality and suggest refinements to our theory.

[21]No existing model of natural language use does this. Some, like Cohen and Levesque's [9], include sophisticated accounts of how the knowledge and intentions of agents are involved in speech acts, but ignore indexicality, in both utterances and mental attitudes. Others, such as Barwise and Perry's [6], include elaborate treatments of indexicality, but do not provide any kind of computational account of how an agent can draw inferences from his prior knowledge and the indexical representations that result from interpreting an utterance.

A Additional Proofs

A.1 Proof of proposition 4.7

First, let us formally state one of the frame assumptions for $\textsc{forward}$ mentioned in section 4.3:

Assumption A.1 ($\textsc{forward}$ does not affect selfori)

$$\models \forall o(\mathbf{selfori} = o \supset \mathbf{AfterNec}(\textsc{forward}, \mathbf{selfori} = o))$$

We prove proposition 4.7 by first showing a more general result, lemma A.2 below. The following lemma is used in its proof.

Lemma A.1

$\models \forall p \forall o(\textsc{MovedToBy}(p, o, \delta) \supset$
$\qquad \mathbf{AfterNec}(\textsc{forward}, \textsc{MovedToBy}(p + \langle 1, 0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))))$

Proof:
1. Take arbitrary $\mathsf{e} = \langle \mathsf{w}, \mathsf{a}, \mathsf{t}\rangle$ and g. Assume that $\mathsf{e}, \mathsf{g} \models \textsc{MovedToBy}(p, o, \delta)$.
2. Take arbitrary w^* and t_e such that $\mathsf{w} \approx_{\mathsf{t}} \mathsf{w}^*$ and $\mathsf{t} \preceq \mathsf{t}_e$. Let $\mathsf{e}^* = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}\rangle$ and $\mathsf{e}^*_e = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_e\rangle$. Assume that $\Delta([\![\textsc{forward}]\!]_{\mathsf{e}^*,\mathsf{g}}, \mathsf{e}^*, \mathsf{t}_e)$.
3. By proposition 3.7, 1 implies that $\mathsf{e}^*, \mathsf{g} \models \textsc{MovedToBy}(p, o, \delta)$.
4. By this and 2, it follows that there exists t_s such that

$$\langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s\rangle, \mathsf{g}\{t_e \to \mathsf{t}_e\} \models \mathbf{Does}((\delta; \textsc{forward}), t_e)$$

Let $\mathsf{e}^*_s = \langle \mathsf{w}^*, \mathsf{a}, \mathsf{t}_s\rangle$. We must now show that $[\![\mathbf{selfori}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\textsc{roriToAori}(o)]\!]_{\mathsf{e}^*_s,\mathsf{g}}$ and

$$[\![\mathbf{here}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\textsc{rposToApos}(p + \langle 1, 0\rangle \times \textsc{rot}(o))]\!]_{\mathsf{e}^*_s,\mathsf{g}}.$$

5. By assumption A.1 and the reflexivity of $\approx_{\mathsf{t}}$ (constraint 3.4), we must have that $[\![\mathbf{selfori}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\mathbf{selfori}]\!]_{\mathsf{e}^*,\mathsf{g}}$. By 3, we have that $[\![\mathbf{selfori}]\!]_{\mathsf{e}^*,\mathsf{g}} = [\![\textsc{roriToAori}(o)]\!]_{\mathsf{e}^*_s,\mathsf{g}}$. So it must be the case that $[\![\mathbf{selfori}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\textsc{roriToAori}(o)]\!]_{\mathsf{e}^*_s,\mathsf{g}}$.
6. By assumption 4.4 and the reflexivity of $\approx_{\mathsf{t}}$, we have that

$$[\![\mathbf{here}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\mathbf{here}]\!]_{\mathsf{e}^*,\mathsf{g}} + \langle 1, 0\rangle \times \textsc{rot}([\![\mathbf{selfori}]\!]_{\mathsf{e}^*,\mathsf{g}})$$

By 3, we have that $[\![\mathbf{here}]\!]_{\mathsf{e}^*,\mathsf{g}} = [\![\textsc{rposToApos}(p)]\!]_{\mathsf{e}^*_s,\mathsf{g}}$, which implies that

$$[\![\mathbf{here}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\textsc{rposToApos}(p)]\!]_{\mathsf{e}^*_s,\mathsf{g}} + (\langle 1, 0\rangle \times \textsc{rot}([\![\mathbf{selfori}]\!]_{\mathsf{e}^*,\mathsf{g}}))$$

By 5 we then get that

$$[\![\mathbf{here}]\!]_{\mathsf{e}^*_e,\mathsf{g}} = [\![\textsc{rposToApos}(p)]\!]_{\mathsf{e}^*_s,\mathsf{g}} + (\langle 1, 0\rangle \times \textsc{rot}([\![\textsc{roriToAori}(o)]\!]_{\mathsf{e}^*_s,\mathsf{g}}))$$

By the definitions of $\textsc{rposToApos}$ and $\textsc{roriToAori}$, this means that

$$[\![\mathbf{here}]\!]_{\mathsf{e}_e^*,\mathsf{g}} = [\![(\mathbf{here} + p \times \textsc{rot}(\mathbf{selfori})) + \langle 1,0\rangle \times \textsc{rot}(\textsc{mod}_{2\pi}(\mathbf{selfori} + o))]\!]_{\mathsf{e}_s^*,\mathsf{g}}$$

Thus

$$\begin{aligned}[\![\mathbf{here}]\!]_{\mathsf{e}_e^*,\mathsf{g}} &= [\![\mathbf{here} + (p + \langle 1,0\rangle \times \textsc{rot}(o)) \times \textsc{rot}(\mathbf{selfori})]\!]_{\mathsf{e}_s^*,\mathsf{g}} \\ &= [\![\textsc{rposToApos}(p + \langle 1,0\rangle \times \textsc{rot}(o))]\!]_{\mathsf{e}_s^*,\mathsf{g}}\end{aligned}$$

7. 4, 5 and 6 imply that $\mathsf{e}_e^*, \mathbf{g} \models \textsc{MovedToBy}(p + \langle 1,0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))$. Since e, w^*, $\mathbf{t}_e$, and $\mathbf{g}$ are arbitrary, this together with 2 implies that

$$\models \mathbf{AfterNec}(\textsc{forward}, \textsc{MovedToBy}(p + \langle 1,0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward})))$$

■

Lemma A.2

$\models \forall p \forall o(\mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset$
$\quad \mathbf{Can}(\textsc{forward}, \textsc{MovedToBy}(p + \langle 1,0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))))$

Proof:
Let $\varphi \overset{\text{def}}{=} \textsc{MovedToBy}(p, o, \delta)$ and

$$\varphi' \overset{\text{def}}{=} \textsc{MovedToBy}(p + \langle 1,0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))$$

Take arbitrary e and $\mathbf{g}$. Suppose that $\mathsf{e}, \mathbf{g} \models \mathbf{Know}(\varphi)$. We must show that $\mathsf{e}, \mathbf{g} \models \mathbf{Can}(\textsc{forward}, \varphi')$. By assumption 4.2, we must have that $\mathsf{e}, \mathbf{g} \models \exists d \mathbf{Know}(d = \textsc{forward})$. It is easy to show that assumption 4.4 implies that $\models \mathbf{PhyPoss}(\textsc{forward})$. Thus by proposition 3.3, $\models \mathbf{Know}(\mathbf{PhyPoss}(\textsc{forward}))$. By propositions 3.3 and 3.2, lemma A.1 implies that $\models \mathbf{Know}(\varphi) \supset \mathbf{Know}(\mathbf{AfterNec}(\textsc{forward}, \varphi'))$. By the above supposition, this implies that $\mathsf{e}, \mathsf{g} \models \mathbf{Know}(\mathbf{AfterNec}(\textsc{forward}, \varphi'))$. ■

Let us now proceed with the proof of proposition 4.7 proper. It is easy to show that $\models \textsc{MovedToBy}(\langle 0,0\rangle, 0, \mathbf{noOp})$. By proposition 3.3, we must then also have that $\models \mathbf{Know}(\textsc{MovedToBy}(\langle 0,0\rangle, 0, \mathbf{noOp}))$. Thus, by lemma A.2, we have that

$$\models \mathbf{Can}(\textsc{forward}, \textsc{MovedToBy}(\langle 0,0\rangle + \langle 1,0\rangle \times \textsc{rot}(0), 0, (\mathbf{noOp}; \textsc{forward})))$$

Since $\models \mathbf{Does}((\mathbf{noOp}; \delta), \theta^t) \equiv \mathbf{Does}(\delta, \theta^t)$, the desired result follows by proposition 4.6 and assumption 4.8. ■

A.2 Proof of proposition 4.8

As for proposition 4.7, we proceed by first showing a more general result:

Lemma A.3

For all $\mathsf{n} \in \mathbb{N}$,
$\models \forall p \forall o(\mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset$
$\quad \mathbf{Can}(\textsc{forward}^{\mathsf{n}}, \textsc{MovedToBy}(p + \langle \mathsf{n}, 0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}^{\mathsf{n}}))))$

Proof: By induction over n.
Base case: $\mathsf{n} = 0$. In this case we only need to show that

$\models \forall p \forall o(\mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset$
$\quad \mathbf{Know}(\textsc{MovedToBy}(p + \langle 0, 0\rangle \times \textsc{rot}(o), o, (\delta; \mathbf{noOp}))))$

It is easy to show that $\models \mathbf{Does}((\delta; \mathbf{noOp}), \theta^t) \equiv \mathbf{Does}(\delta, \theta^t)$. This together with proposition 4.6 and assumption 4.8 implies that

$$\models \textsc{MovedToBy}(p + \langle 0, 0\rangle \times \textsc{rot}(o), o, (\delta; \mathbf{noOp})) \equiv \textsc{MovedToBy}(p, o, \delta)$$

From this, the desired result follows easily by propositions 3.3 and 3.2.
Induction step: Assume that the formula is valid for n and show that it must then be valid for $\mathsf{n} + 1$. By the induction hypothesis and proposition 4.6, we have that

$\models \mathbf{Know}(\textsc{MovedToBy}(p + \langle 1, 0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))) \supset$
$\quad \mathbf{Can}(\textsc{forward}^{\mathsf{n}}, \textsc{MovedToBy}($
$\qquad p + \langle 1, 0\rangle \times \textsc{rot}(o) + \langle \mathsf{n}, 0\rangle \times \textsc{rot}(o), o, ((\delta; \textsc{forward}); \textsc{forward}^{\mathsf{n}})))$

It is easy to show that $\models \mathbf{Does}((\delta_1; \delta_2); \delta_3) \equiv \mathbf{Does}(\delta_1; (\delta_2; \delta_3))$. By this, proposition 4.6, and assumption 4.8, the above simplifies to

$\models \mathbf{Know}(\textsc{MovedToBy}(p + \langle 1, 0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}))) \supset$
$\quad \mathbf{Can}(\textsc{forward}^{\mathsf{n}}, \textsc{MovedToBy}(p + \langle \mathsf{n} + 1, 0\rangle \times \textsc{rot}(o), o, (\delta; \textsc{forward}^{\mathsf{n}+1})))$

By propositions 3.8 and 3.9, it follows from this and lemma A.2 that the formula is valid for $\mathsf{n} + 1$. ■

Given this lemma, we can prove proposition 4.8 in the same way that proposition 4.7 was proved in the previous subsection (lemma A.3 is used instead of lemma A.2). ■

A.3 Proof of proposition 4.10

In this case too, we proceed by first showing a more general result, lemma A.5 below. The proof uses the following strengthened version of proposition 4.9:

Lemma A.4

For all $\mathsf{n} \in \mathbb{N}$,
$\models \forall p \forall o(\mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset$
$\quad\quad \mathbf{Can}(\textsc{left}^{\mathsf{n}}, \textsc{MovedToBy}(p, \textsc{mod}_{2\pi}(o + \mathsf{n}\pi/2), (\delta; \textsc{left}^{\mathsf{n}}))))$

We omit its proof, which goes along the same lines as that of lemma A.3.

Lemma A.5

For all $\mathsf{n}, \mathsf{m} \in \mathbb{Z}$,
$\models \forall p \forall o(\mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset$
$\quad \mathbf{Can}(\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m} \rangle), \textsc{MovedToBy}(p + \langle \mathsf{n}, \mathsf{m} \rangle \times \textsc{rot}(o), o, (\delta; \textsc{goRpos}(\langle \mathsf{n}, \mathsf{m} \rangle)))))$

Proof:
Case 1 — $\mathsf{m} = 0$ and $\mathsf{n} \in \mathbb{N}$: Then $\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m} \rangle) = \textsc{forward}^{\mathsf{n}}$ and the desired result follows immediately from lemma A.3.
Case 2 — $\mathsf{m} = 0$ and $\mathsf{n} < 0$: Then $\textsc{goRpos}(\langle \mathsf{n}, \mathsf{m} \rangle) = \textsc{left}^2; \textsc{goRpos}(\langle -\mathsf{n}, 0 \rangle); \textsc{left}^2$. By lemma A.4, proposition 4.6, and assumption 4.8, we have that

$$\models \mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset \mathbf{Can}(\textsc{left}^2, \textsc{MovedToBy}(p, \textsc{mod}_{2\pi}(o + \pi), (\delta; \textsc{left}^2)))$$

By case 1 and proposition 4.6, we have that

$$\begin{aligned}&\models \mathbf{Know}(\textsc{MovedToBy}(p, \textsc{mod}_{2\pi}(o + \pi), (\delta; \textsc{left}^2))) \supset \\ &\quad \mathbf{Can}(\textsc{goRpos}(\langle -\mathsf{n}, 0 \rangle), \textsc{MovedToBy}(p + \langle -\mathsf{n}, 0 \rangle \times \textsc{rot}(\textsc{mod}_{2\pi}(o + \pi)), \\ &\qquad\qquad \textsc{mod}_{2\pi}(o + \pi), (\delta; \textsc{left}^2; \textsc{goRpos}(\langle -\mathsf{n}, 0 \rangle)))))\end{aligned}$$

By propositions 3.8 and 3.9, one can chain these results to obtain:

$$\begin{aligned}&\models \mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset \\ &\quad \mathbf{Can}((\textsc{left}^2; \textsc{goRpos}(\langle -\mathsf{n}, 0 \rangle)), \\ &\qquad \textsc{MovedToBy}(p + \langle -\mathsf{n}, 0 \rangle \times \textsc{rot}(\textsc{mod}_{2\pi}(o + \pi)), \\ &\qquad\qquad \textsc{mod}_{2\pi}(o + \pi), (\delta; \textsc{left}^2; \textsc{goRpos}(\langle -\mathsf{n}, 0 \rangle)))))\end{aligned}$$

By chaining this again with an appropriate instance of lemma A.4, one obtains that

$$\begin{aligned}&\models \mathbf{Know}(\textsc{MovedToBy}(p, o, \delta)) \supset \\ &\quad \mathbf{Can}(\textsc{goRpos}(\langle \mathsf{n}, 0 \rangle), \textsc{MovedToBy}(p + \langle -\mathsf{n}, 0 \rangle \times \textsc{rot}(\textsc{mod}_{2\pi}(o + \pi)), \\ &\qquad\qquad \textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o + \pi) + \pi), (\delta; \textsc{goRpos}(\langle \mathsf{n}, 0 \rangle)))))\end{aligned}$$

Now it is easy to check that

$$\begin{aligned}\langle -\mathsf{n}, 0 \rangle \times \textsc{rot}(\textsc{mod}_{2\pi}(o + \pi)) &= \langle -\mathsf{n}\cos(o + \pi), -\mathsf{n}\sin(o + \pi) \rangle \\ &= \langle \mathsf{n}\cos(o), \mathsf{n}\sin(o) \rangle \\ &= \langle \mathsf{n}, 0 \rangle \times \textsc{rot}(o)\end{aligned}$$

It is also the case that

$$\begin{aligned}\textsc{roriToAori}(\textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o+\pi)+\pi)) &= \textsc{mod}_{2\pi}(\mathbf{selfori}+\textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o+\pi)+\pi))\\ &= \textsc{mod}_{2\pi}(\mathbf{selfori}+o)\\ &= \textsc{roriToAori}(o)\end{aligned}$$

So by the definition of MovedToBy and proposition 4.8, we must have that

$$\begin{aligned}&\models\textsc{MovedToBy}(p+\langle -\mathsf{n},0\rangle\times\textsc{rot}(\textsc{mod}_{2\pi}(o+\pi)),\textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o+\pi)+\pi),\\ &\qquad(\delta;\textsc{goRpos}(\langle\mathsf{n},0\rangle)))\equiv\textsc{MovedToBy}(p+\langle\mathsf{n},0\rangle\times\textsc{rot}(o),o,(\delta;\textsc{goRpos}(\langle\mathsf{n},0\rangle)))\end{aligned}$$

The desired result follows.
Case 3 — $\mathsf{m}\neq 0$: Then $\textsc{goRpos}(\langle\mathsf{n},\mathsf{m}\rangle)=\textsc{goRpos}(\langle\mathsf{n},0\rangle);\textsc{left};\textsc{goRpos}(\langle\mathsf{m},0\rangle);\textsc{left}^3$. The proof is similar to case 2. By chaining (propositions 3.8 and 3.9) appropriate instances of case 2 with instances of lemma A.4, we show that

$$\begin{aligned}&\models\mathbf{Know}(\textsc{MovedToBy}(p,o,\delta))\supset\\ &\quad\mathbf{Can}(\textsc{goRpos}(\langle\mathsf{n},\mathsf{m}\rangle),\\ &\qquad\quad\textsc{MovedToBy}(p+\langle\mathsf{n},0\rangle\times\textsc{rot}(o)+\langle\mathsf{m},0\rangle\times\textsc{rot}(\textsc{mod}_{2\pi}(o+\pi/2)),\\ &\qquad\qquad\qquad\textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o+\pi/2)+3\pi/2),(\delta;\textsc{goRpos}(\langle\mathsf{n},\mathsf{m}\rangle))))\end{aligned}$$

One then shows that

$$\begin{aligned}&\models\textsc{MovedToBy}(p+\langle\mathsf{n},0\rangle\times\textsc{rot}(o)+\langle\mathsf{m},0\rangle\times\textsc{rot}(\textsc{mod}_{2\pi}(o+\pi/2)),\\ &\qquad\qquad\textsc{mod}_{2\pi}(\textsc{mod}_{2\pi}(o+\pi/2)+3\pi/2),(\delta;\textsc{goRpos}(\langle\mathsf{n},\mathsf{m}\rangle)))\\ &\quad\equiv\textsc{MovedToBy}(p+\langle\mathsf{n},\mathsf{m}\rangle\times\textsc{rot}(o),o,(\delta;\textsc{goRpos}(\langle\mathsf{n},\mathsf{m}\rangle)))\end{aligned}$$

to obtain the desired result. ■

Given this lemma, we can then prove proposition 4.10 in the same way that proposition 4.7 was proved in the section A.1 (lemma A.5 is used instead of lemma A.2). ■

A.4 Proof of proposition 4.13

Let us first introduce some shorthands:

- $\mathrm{UOAT}(\theta_r^i)\overset{\text{def}}{=}\exists v^i(\textsc{Object}(v^i)\wedge\textsc{rpos}(v^i)=\langle\theta_r^i,0\rangle\wedge\neg\textsc{Holding}(v^i))$, provided v^i does not occur free in θ_r^i
- $\mathrm{UOH}\overset{\text{def}}{=}\mathrm{UOAT}(0)$
- $\mathrm{UOBTW}(\theta_l^i,\theta_u^i)\overset{\text{def}}{=}\exists v^i(\theta_l^i\leq v^i\leq\theta_u^i\wedge\mathrm{UOAT}(v^i))$, provided v^i does not occur free in θ_l^i and θ_u^i

UOAT(θ_r^i) says that there is an unheld object θ_r^i squares directly in front of the agent (behind the agent if θ_r^i is negative); UOH says that there is an unheld object **here**; and finally, UOBTW(θ_l^i, θ_u^i) says that there is an unheld object between θ_l^i and θ_u^i squares directly in front of or behind the agent.

We prove proposition 4.13 by first establishing a result concerning its scanning component, lemma A.8 below. This result depends on two sublemmas. The first concerns the FORWARD action:

Lemma A.6

For all $\mathsf{n} \in \mathbb{N}$, $\models \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, 0)) \supset \mathbf{Can}(\textsc{forward}, \neg\mathrm{UOBTW}(-\mathsf{n} - 1, -1))$

This lemma can be proven from assumptions 4.2 and 4.4, as well as the frame assumptions for FORWARD mentioned in section 4.3; a detailed proof of a closely related result appears in [29]. The second preliminary lemma concerns the SENSE action:

Lemma A.7

For all $\mathsf{n} \in \mathbb{N}$,
$\models \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, -1)) \supset \mathbf{Can}(\textsc{sense}, \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, 0)))$

It can be proven from assumptions 4.2 and 4.10, and the frame assumptions for SENSE mentioned in sections 4.4 and 4.3.

Let us now proceed with the main lemma concerned with scanning:

Lemma A.8

For all $\mathsf{k}, \mathsf{n} \in \mathbb{N}$,
$$\models \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, 0)) \supset$$
$$\mathbf{Can}(\textsc{scan}_{\mathsf{k}}(\mathrm{UOH}), \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n} - \mathsf{k}, 0)))$$

Proof: By induction over k.
Base case: $\textsc{scan}_0(\mathrm{UOH}) \stackrel{\mathrm{def}}{=} \mathbf{noOp}$ and $\models \mathbf{Can}(\mathbf{noOp}, \varphi) \equiv \mathbf{Know}(\varphi)$, so all we need to show is that for all $\mathsf{n} \in \mathbb{N}$,

$$\models \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, 0)) \supset$$
$$\mathbf{Know}(\mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n}, 0)))$$

This is easily done using propositions 3.5, 3.3, and 3.2.
Induction step: Assume that the formula is valid for k and show that it must then be valid for $\mathsf{k}+1$. $\textsc{scan}_{\mathsf{k}+1}(\mathrm{UOH}) \stackrel{\mathrm{def}}{=} \mathbf{if}(\neg\mathrm{UOH}, \textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH}), \mathbf{noOp})$, and so by proposition 3.10, we have that

$$\begin{aligned}&\models \mathbf{Can}(\textsc{scan}_{\mathsf{k}+1}(\mathrm{UOH}), \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0))) \equiv\\ &\quad (\mathbf{Know}(\mathrm{UOH}) \wedge \mathbf{Can}(\mathbf{noOp}, \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0)))) \vee\\ &\quad (\mathbf{Know}(\neg\mathrm{UOH}) \wedge \mathbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH}),\\ &\qquad\qquad \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg\mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0))))\end{aligned}$$

By propositions 3.5, 3.3, and 3.2, it is easy to show that for all $\mathsf{n} \in \mathbb{N}$,

$$\begin{aligned}&\models \mathbf{Know}(\mathrm{UOH}) \supset \\ &\quad \mathbf{Know}(\mathrm{UOH}) \wedge \mathbf{Can}(\mathbf{noOp}, \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0)))\end{aligned}$$

What remains to be proved is that for all $\mathsf{n} \in \mathbb{N}$,

$$\begin{aligned}\models \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n}, 0)) \supset (&\mathbf{Know}(\neg \mathrm{UOH}) \wedge \mathbf{Can}(\textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH}), \\ &\mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0))))\end{aligned}$$

Clearly $\models \neg \mathrm{UOBTW}(-\mathsf{n}, 0) \supset \neg \mathrm{UOH}$ for $\mathsf{n} \in \mathbb{N}$; so by propositions 3.3 and 3.2, we have that for all $\mathsf{n} \in \mathbb{N}$, $\models \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n}, 0)) \supset \mathbf{Know}(\neg \mathrm{UOH})$. From lemma A.7 and the induction hypothesis, it follows by proposition 3.8 that for all $\mathsf{n} \in \mathbb{N}$,

$$\begin{aligned}&\models \mathbf{Know}(\neg \mathrm{UOBTW}(-(\mathsf{n}+1), -1)) \supset \\ &\quad \mathbf{Can}(\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH}), \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg \mathrm{UOBTW}(-(\mathsf{n}+1) - \mathsf{k}, 0)))\end{aligned}$$

By propositions 3.8 and 3.9, this and lemma A.6 imply that for all $\mathsf{n} \in \mathbb{N}$,

$$\begin{aligned}\models \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n}, 0)) \supset \mathbf{Can}(&\textsc{forward}; \textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH}), \\ &\mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n} - (\mathsf{k}+1), 0))))\end{aligned}$$

■

We can now complete the proof of proposition 4.13. By lemmas A.8 and A.7 and proposition 3.8, we must have that for all $\mathsf{n} \in \mathbb{N}$,

$$\begin{aligned}&\models \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n}, -1)) \supset \\ &\quad \mathbf{Can}((\textsc{sense}; \textsc{scan}_{\mathsf{k}}(\mathrm{UOH})), \mathbf{Know}(\mathrm{UOH}) \vee \mathbf{Know}(\neg \mathrm{UOBTW}(-\mathsf{n} - \mathsf{k}, 0)))\end{aligned}$$

Clearly $\models \neg \mathrm{UOBTW}(0, -1)$, and thus by proposition 3.3 $\models \mathbf{Know}(\neg \mathrm{UOBTW}(0, -1))$; the desired result follows. ■

Acknowledgements

This paper describes a revised version of work that was part of the first author's Ph.D. thesis at the Dept. of Computer Science, University of Toronto, which was supervised by the second author (jointly with Graeme Hirst). Revisions were done while the first author was at NTT Basic Research Laboratories in Musashino, Japan and the School of Computing Science, Simon Fraser University, Burnaby, Canada. This work was funded by the Institute for Robotics and Intelligent Systems and the Natural Science and Engineering Research Council of Canada, and the Information Technology Research Centre of Ontario. David Israel, Graeme Hirst, and the referees have provided many valuable suggestions. Discussions with Phil Agre, Adam Grove, Joe Halpern, and Joe Nunes have also been helpful.

References

[1] Philip E. Agre. Review of [54]. *Artificial Intelligence*, 43(3):369–384, 1990.

[2] Philip E. Agre and David Chapman. Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272, Seattle, WA, July 1987. American Association for Artificial Intelligence, Morgan Kaufmann Publishing.

[3] Philip E. Agre and David Chapman. What are plans for? *Robotics and Autonomous Systems*, 6:17–34, 1990.

[4] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms.* Addison-Wesley Publishing, Reading, MA, 1974.

[5] Jon Barwise. Information and circumstance. *Notre Dame Journal of Formal Logic*, 27(3):324–338, 1986.

[6] Jon Barwise and John Perry. *Situations and Attitudes.* MIT Press/Bradford Books, Cambridge, MA, 1983.

[7] Hector-Neri. Castañeda. On the logic of attributions of self-knowledge to others. *Journal of Philosophy*, 65(15):439–456, 1968.

[8] Brian F. Chellas. *Modal Logic: An Introduction.* Cambridge University Press, Cambridge, UK, 1980.

[9] Philip R. Cohen and Hector J. Levesque. Rational interaction as the basis for communication. In Philip R. Cohen, J. Morgan, and Martha E. Pollack, editors, *Intentions in Communication*, pages 221–255. MIT Press, Cambridge, MA, 1990.

[10] Ernest Davis. Inferring ignorance from the locality of visual perception. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 786–791, St. Paul, MN, August 1988. American Association for Artificial Intelligence.

[11] Ernest Davis. Reasoning about hand-eye coordination. In *Working Notes, IJCAI Workshop on Knowledge, Perception, and Planning*, pages 1–6, Detroit, MI, August 1989.

[12] Ernest Davis. Solutions to a paradox of perception with limited acuity. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 79–82, Toronto, ON, May 1989. Morgan Kaufmann Publishers.

[13] Robert Goldblatt. *Logics of Time and Computation.* CSLI Lecture Notes No. 7. Center for the Study of Language and Information, Stanford University, Stanford, CA, 1987.

[14] Adam J. Grove. *Topics in Multi-Agent Epistemic Logic.* PhD thesis, Department of Computer Science, Stanford University, Stanford, CA, October 1992.

[15] Adam J. Grove and Joseph Y. Halpern. Naming and identity in a multi-agent epistemic logic. In James Allen, Richard Fikes, and Erik Sandewall, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, pages 301–312, Cambridge, MA, 1991. Morgan Kaufmann Publishing.

[16] Andrew R. Haas. A syntactic theory of belief and action. *Artificial Intelligence*, 28:245–292, 1986.

[17] Andrew R. Haas. Indexical expressions and planning. Unpublished manuscript, Department of Computer Science, State University of New York, Albany, NY, 1991.

[18] Joseph Y. Halpern and Yoram Moses. A guide to the modal logics of knowledge and belief: Preliminary draft. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 480–490, Los Angeles,CA, August 1985. Morgan Kaufmann Publishing.

[19] Jaakko Hintikka. *Knowledge and Belief.* Cornell University Press, Ithaca, NY, 1962.

[20] Jaakko Hintikka. Semantics for the propositional attitudes. In J. W. Davis, D. J. Hockney, and K. W. Wilson, editors, *Philosophical Logic*, pages 21–45. D. Reidel Publishing, Dordrecht, Holland, 1969.

[21] G. E. Hughes and M. J. Cresswell. *An Introduction to Modal Logic.* Methuen, London, UK, 1968.

[22] David J. Israel. The role of propositional objects of belief in action. Technical Report CSLI-87-72, CSLI, Stanford University, Stanford, CA, January 1987.

[23] Leslie P. Kaelbling. An architecture for intelligent systems. In *Reasoning about Actions and Plans: Proceedings of the 1986 Workshop*, pages 395–410, Timberline, OR, July 1987. Morgan Kaufmann Publishing.

[24] Leslie P. Kaelbling and Stanley J. Rosenschein. Action and planning in embedded agents. *Robotics and Autonomous Systems*, 6:35–48, 1990.

[25] Kurt Konolige. A first order formalization of knowledge and action for a multi-agent planning system. In J.E. Hays and D. Michie, editors, *Machine Intelligence*, volume 10. Ellis Horwood, Chichester, UK, 1982.

[26] Kurt Konolige. *A Deduction Model of Belief.* Pitman Publishing, 1986.

[27] Saul A. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[28] Amichai Kronfeld. *Reference and Computation : An Essay in Applied Philosophy of Language.* Cambridge University Press, New York, NY, 1990.

[29] Yves Lespérance. *A Formal Theory of Indexical Knowledge and Action.* PhD thesis, Department of Computer Science, University of Toronto, Toronto, ON, January 1991. Also published as technical report CSRI-248.

[30] Yves Lespérance. An approach to modeling indexicality in action and communication. In John Horty and Yoav Shoham, editors, *Reasoning about Mental States: Formal Theories & Applications, Papers from the 1993 AAAI Spring Symposium*, pages 79–85, Stanford, CA, March 1993. Technical Report SS-93-05, AAAI Press. Also appears in *Proceedings of the IJCAI Workshop on Using Knowledge in its Context*, Chambery, France, August 1993.

[31] Yves Lespérance and Hector J. Levesque. Indexical knowledge in robot plans. In *Proceedings of the Eight National Conference on Artificial Intelligence*, pages 868–874, Boston, August 1990. American Association for Artificial Intelligence, AAAI Press/MIT Press.

[32] Yves Lespérance and Hector J. Levesque. An argument for indexical representations in temporal reasoning. Submitted to the Canadian Artificial Intelligence Conference (AI'94), November 1993.

[33] Yves Lespérance and Hector J. Levesque. On the logic of indexical knowledge and action. In preparation, 1994.

[34] Yves Lespérance, Hector J. Levesque, and Fangzhen Lin. A formalization of ability and knowing how that avoids the frame problem. Submitted to the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94), November 1993.

[35] Hector J. Levesque. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23:155–212, 1984.

[36] Hector J. Levesque. A logic of implicit and explicit belief. In *Proceedings of the National Conference on Artificial Intelligence*, pages 198–202, Austin, TX, 1984. American Association for Artificial Intelligence.

[37] David Lewis. Attitudes *de dicto* and *de se*. *The Philosophical Review*, 88(4):513–543, 1979.

[38] John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press, Edinburgh, UK, 1979.

[39] Elliott Mendelson. *Introduction to Mathematical Logic (Second Edition).* D. Van Nostrand Co., New York, 1979.

[40] Robert C. Moore. Reasoning about knowledge and action. Technical Report 191, AI Center, SRI International, Menlo Park, CA, October 1980.

[41] Robert C. Moore. A formal theory of knowledge and action. In J. R. Hobbs and Robert C. Moore, editors, *Formal Theories of the Common Sense World*, pages 319–358. Ablex Publishing, Norwood, NJ, 1985.

[42] Leora Morgenstern. Knowledge preconditions for actions and plans. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 867–874, Milan, Italy, August 1987. Morgan Kaufmann Publishing.

[43] Leora Morgenstern. *Foundations of a Logic of Knowledge, Action, and Communication.* PhD thesis, Department of Computer Science, New York University, New York, NY, 1988.

[44] Leora Morgenstern. Knowledge and the frame problem. In K. Ford and P. Hayes, editors, *Reasoning Agents in a Dynamic World: The Frame Problem.* JAI Press, Greenwich, 1991.

[45] A. Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

[46] José H. T. Nunes and Hector J. Levesque. Ability and commitment. Unpublished manuscript, Department of Computer Science, University of Toronto, Toronto, ON, 1990.

[47] John Perry. The problem of the essential indexical. *Noûs*, 13:3–21, 1979.

[48] Raymond Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380. Academic Press, San Diego, CA, 1991.

[49] Stanley J. Rosenschein and Leslie P. Kaelbling. The synthesis of digital machines with provable epistemic properties. In Joseph Y. Halpern, editor, *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the 1986 Conference*, pages 83–98, Monterey, CA, 1986. Morgan Kaufmann Publishing.

[50] Richard B. Scherl and Hector J. Levesque. The frame problem and knowledge-producing actions. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 689–695, Washington, DC, July 1993. AAAI Press/The MIT Press.

[51] Richard B. Scherl, Hector J. Levesque, and Yves Lespérance. Indexicals in the situation calculus. In preparation, 1994.

[52] Brian Cantwell Smith. The owl and the electric encyclopedia. *Artificial Intelligence*, 47:251–288, 1991.

[53] Devika Subramanian and John Woodfill. Making the situation calculus indexical. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, pages 467–474, Toronto, ON, May 1989. Morgan Kaufmann Publishing.

[54] Lucy A. Suchman. *Plans and Situated Action: The Problem of Human–Machine Communication.* Cambridge University Press, Cambridge, UK, 1987.

[55] Bonnie Webber, Norman Badler, F. Breckenridge Baldwin, Welton Becket, Barbara Di Eugenio, Christopher Geib, Moon Jung, Libby Levison, Michael Moore, and Michael White. Doing what you're told: Following task instructions in changing, but hospitable environments. Technical Report MS-CIS-92-74, LINC LAB 236, Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA, September 1992. Submitted to *Artificial Intelligence*, special issue on Computational Theories of Interaction and Agency.