Conditional Planning for a Mobile Robot*

Giuseppe De Giacomo, Luca Iocchi, Daniele Nardi, Riccardo Rosati
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
<degiacomo,iocchi,nardi,rosati>@dis.uniroma1.it

Abstract

We present an attempt to reconcile the theoretical work on reasoning about action with the realization of agents, in particular mobile robots. Specifically, we present a logical framework for representing dynamic systems based on description logics, which allows for the formalization of sensing actions. We address the generation of conditional plans by defining a suitable reasoning method in which a plan is extracted from a constructive proof of a query expressing a given goal. We also present an implementation of such a logical framework, which has been tested on the mobile robot "Tino".

1 Introduction

In recent years there has been an attempt to reconcile the theoretical work on reasoning about action with the realization of agents, in particular mobile robots. This field of research has been referred to as *Cognitive Robotics* [10].

A mobile robot can indeed be regarded as an intelligent agent, that is designed both to achieve high-level goals and to be able to promptly react and adjust its behavior based on the information acquired through the sensors. Reactive capabilities are necessary to cope with the uncertainties of the real-world; action planning is important as well, if the robot is faced with situations where the knowledge of the environment is incomplete, subject to varying constraints. The integration of the two kinds of functionalities mentioned above is a critical issue in the design of intelligent agents.

*Abridged version of "Planning with Sensing for a Mobile Robot" appeared in the Proceedings of the 4th European Conference on Planning (ECP'97).

The work reported in the present paper builds on a previous proposal [3], which provides a formal framework for reasoning about action derived from *dynamic logics* [16] and exploits the correspondence between such logics and *description logics*. A number of features that had been analyzed for description logics have proved useful for reasoning about action. Specifically, we have extended the language with an epistemic operator interpreted in terms of minimal knowledge, that allows us to express the knowledge about actions in such a way that we can effectively address the planning problem. We have implemented our proposal for reasoning about action on the mobile robot "Tino", which belongs to the *Erratic* family [8]. The implementation relies on the reasoning facilities offered by the knowledge representation system CLASSIC.

In this paper we extend the previous proposal with the ability of expressing sensing actions [18, 11, 6], i.e. knowledge producing actions that affect the agent's knowledge, but not the environment. We also extend the implementation with the ability of devising and executing conditional plans. From the point of view of the formalism, a new kind of axioms for sensing actions and the ability of propagating knowledge to successor states (in a controlled way) is introduced. From the point of view of the implementation, we design a more powerful method for devising the plan, and extend the capabilities of the plan execution component and of the underlying control system by providing new behaviors realizing the sensing actions.

The paper is organized as follows. We first recall the basic elements of our approach. We then focus on sensing actions and on the method for devising plans in the presence of this kind of actions. We finally describe the implementation of the new features in our robot "Tino".

2 Epistemic DL-based framework for representing actions

Our general framework for representing dynamic systems was originally proposed in [3]. It follows the lines of Rosenschein's work [16], based on propositional dynamic logics (PDLs) [9], and it makes use of the tight correspondence between PDLs and description logics (DLs) [19, 4] that allows for considering PDLs and DLs as notational variants of each other. We use the notation of DLs, focusing on the well-known DL $\mathcal{ALC}$, corresponding to the standard PDL with atomic programs only. Table 1 summarizes the syntax and the semantics of $\mathcal{ALC}$ and the corresponding PDL. In addition, we use

Constructs	PDLs	DLs	DL Semantics
Atm. concept/prop.	A	A	$A^{\mathcal{I}} \subseteq \Delta$
Atm. role/action	R	R	$R^{\mathcal{I}} \subseteq \Delta \times \Delta$
Named individual	—	s	$s^{\mathcal{I}} \in \Delta$
True	$\mathtt{tt}$	$\top$	Δ
False	$\mathtt{ff}$	$\bot$	$\emptyset$
Conjunction	$C \wedge D$	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
Disjunction	$C \vee D$	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
Negation	$\neg C$	$\neg C$	$\Delta \setminus C^{\mathcal{I}}$
Universal quant.	$[R]C$	$\forall R.C$	$\{d \in \Delta \mid \forall d'. (d,d') \in R^{\mathcal{I}} \Rightarrow d' \in C^{\mathcal{I}}\}$
Existential quant.	$\langle R \rangle C$	$\exists R.C$	$\{d \in \Delta \mid \exists d'. (d,d') \in R^{\mathcal{I}} \wedge d' \in C^{\mathcal{I}}\}$
Inclusion assertion	$C \Rightarrow D$	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every $\mathcal{I}$
Instance assertion	—	$C(s)$ $R(s_1, s_2)$	$s^{\mathcal{I}} \in C^{\mathcal{I}}$ $(s_1^{\mathcal{I}}, s_2^{\mathcal{I}}) \in R^{\mathcal{I}}$ for every $\mathcal{I}$

Table 1: Description logic $\mathcal{ALC}$

the two nonmonotonic modal operators: a *minimal knowledge operator* **K** and a *default assumption operator* **A**. These are interpreted according to the nonmonotonic modal logic *MKNF* [12], and give rise to the so-called autoepistemic description logic $\mathcal{ALCK}_{\mathcal{NF}}$ [5]. We do not have the space here to formally introduce such a logical framework, we refer the reader to [3] and [5]. Rather, we give an intuition of the underlying semantics.

The interpretation structures of DLs (PDLs) are essentially *graphs* labeled both on nodes and arcs. *Nodes*, called *individuals* in DLs, (states in PDLs) are labeled by *concepts* (formulae in PDLs) that denote properties of individuals. *Arcs*, called *links* in DL (state transitions in PDLs) are labeled by *roles* (actions in PDLs). Such interpretation structures can be concretely bound to the robot's behavior (possible courses of actions): individuals represent states of the robot and are labeled by concepts representing what is true in that state; links between individuals represent transitions between states of the robot, and are labeled by roles representing the actions that cause the state transition.

However, in general there is not enough information about the robot's environment to model its behavior by means of a single interpretation structure, since the robot's behavior will depend on external circumstances that will be known only at execution time. Rather, we model the robot's behavior with suitable axioms which reflect our (partial) knowledge and which are satisfied by *multiple* interpretation structures. As a consequence, in order to decide which action to perform next the robot can use only those facts that are "valid" in its current state, i.e. that are true in the representative of its current state in all possible interpretation structures. To do so the logical formalism must provide:

- A mechanism to isolate an individual representative of a given robot's state, in each possible interpretation, establishing a one-to-one mapping between the individuals in the different interpretation structures that represent the same robot's state.
- A mechanism to represent that a certain property (concept) is "valid" in a robot's state, i.e. true in the representatives of that state in all possible interpretations.

The minimal knowledge operator $\mathbf{K}$ gives us both the above mechanisms. On the one hand, it allows for isolating the representatives of robot's states in the different structures establishing a one-to-one mapping among them through the so-called *known individuals*. In general, known individuals will be only those that are explicitly *named* in some axiom (in our case, we will have a single such named individual, *init*, denoting the initial state of the robot) and those generated by a special use of $\mathbf{K}$ on roles denoting actions. On the other hand, it allows for denoting the "validity" of a property in a robot's state. In particular, an epistemic implication of the form $\mathbf{K}C \sqsubseteq D$ differs from the non-modal implication $C \sqsubseteq D$ since D is concluded for a given known individual only if C is necessarily true ("valid") for that known individual. This prevents forms of reasoning by cases such as the following: let $\Sigma = \{[C_1 \sqcup C_2](init), \mathbf{K}C_1 \sqsubseteq D, \mathbf{K}C_2 \sqsubseteq D\}$, then $\Sigma \not\models D(init)$, while let $\Sigma' = \{[C_1 \sqcup C_2](init), C_1 \sqsubseteq D, C_2 \sqsubseteq D\}$, then $\Sigma' \models D(init)$. Moreover, for $\mathbf{K}C \sqsubseteq D$ the contrapositive does not hold, i.e. $\neg D$ does not imply $\neg C$. Epistemic sentences $\mathbf{K}C \sqsubseteq D$ can be naturally interpreted in terms of *rules*, i.e. a forward reasoning mechanism.

The default assumption operator $\mathbf{A}$ allows for expressing justifications of default rules [15], and the combined usage of $\mathbf{K}$ and $\mathbf{A}$ allows for formalizing defaults in terms of modal formulas. We use it here in relation with sensing actions in a very specific way (see below).

3 Robot's behavior representation

We distinguish two kinds of robot's actions: *moving actions* and *sensing actions*. Both kinds of actions are considered *deterministic*, in the sense that a unique successor state will be generated by each action. We first focus on moving actions only as in [3], then in the next section we consider sensing actions as well. Here, we call moving actions all the actions that result in a change in the enviroment, like, for example, a change in the position of the robot. Like most approaches to reasoning about action [14, 20] we express our knowledge in terms of a finite set of axioms forming a knowledge base Σ. Such axioms are partitioned in the classes below, each formalized in a specific way.

Static axioms (Γ_S)[1] They are used for representing background knowledge, which is invariant with respect to the execution of actions. Static axioms hold in every state, and they do not depend on actions. We formalize static axioms as $\mathcal{ALC}$ inclusion assertions, not involving action-roles, although in general they can involve other roles used for structuring concept (i.e. property) descriptions and form complex taxonomies of properties that are typical of DLs.

Action precondition axioms (Γ_P) They describe under which circumstances it is possible to execute an action. We formalize action precondition axioms through epistemic sentences of the form:

$$\mathbf{K}C \sqsubseteq \exists \mathbf{K}R_M.\top \tag{1}$$

where C is an $\mathcal{ALC}$ concept and R_M a moving action. This axiom can be read as: if C holds in the (known individual denoting the) current state s, then there exists a (known individual denoting a) state s' which is the R_M-successor of s.

Effect axioms (Γ_E) They specify the effects of executing an action R_M in a state satisfying certain *premises* C. We formalize effect axioms through epistemic sentences of the form:

$$\mathbf{K}C \sqsubseteq \forall \mathbf{K}R_M.\mathbf{K}D \tag{2}$$

where C and D are $\mathcal{ALC}$ concepts. This axiom can be read as: if C holds in the (known individual denoting the) current state s, then for each (known individual denoting an) R-successor s' of s, D holds in s' in all interpretations.[2]

Initial state description axioms (Γ_I) They specify the properties that hold in the initial state of the robot. We formalize them by introducing explicitly a named (and hence known) individual *init* denoting the initial state, and instance assertions of the form:

$$C(init) \tag{3}$$

[1] Sometimes called *domain contraints* or *state constraints.*

[2] Note that, since actions are deterministic, there is at most one R-successor of s.

where C is an $\mathcal{ALC}$ concept. This axiom can be read as: C holds in the state *init* in every possible interpretation.

The use of the $\mathbf{K}$ operator in the antecedent of the action precondition axioms restricts the applicability of an action to those states in which the precondition for the action are known by the agent. In the initial state, the only known properties are those implied by the initial state description axioms and the static axioms; in any other state, the known properties are those implied by the effects of the last executed action (together with the static axioms).

We do not try to address the frame problem for moving actions by enforcing some general form of *common sense inertia law*: if a property C persists after a certain action R_M, the effect axiom $\mathbf{K}C \sqsubseteq \forall \mathbf{K}R_M.\mathbf{K}C$ must be included. Obviously some general default persistence mechanism would be highly desirable and in this paper we devise such a mechanism for the special case of sensing actions (see next section).

Planning problem In deductive planning one is typically interested in answering the following question: "Is there a sequence of actions that, starting from an initial state, leads to a state where a given property (the goal) holds?". This is captured in our framework by the following logical implication:

$$\Sigma \models PLAN\_FOR\_G(init) \tag{4}$$

where: (i) Σ is the knowledge base including the static axioms Γ_S, the action preconditions axioms Γ_P, the effect axioms Γ_E, and the initial state description axioms Γ_I; (ii) $PLAN\_FOR\_G(init)$ denotes that $PLAN\_FOR\_G$ holds in the initial state *init*, where $PLAN\_FOR\_G$ is *any* concept belonging to the set $\mathcal{P}_S$ defined inductively as: $\mathbf{K}G \in \mathcal{P}_S$; if $C \in \mathcal{P}_S$, then $\exists \mathbf{K}R_{M_i}.C \in \mathcal{P}_S$, for every moving action R_{M_i}.[3]

In other words, $PLAN\_FOR\_G$ stands for any concept expression of the form $\exists \mathbf{K}R_{M_1}.\exists \mathbf{K}R_{M_2}.\ldots.\exists \mathbf{K}R_{M_n}.\mathbf{K}G$ in which $n \geq 0$ and each R_{M_i} is a moving action, and it expresses the fact that from the initial state *init* there exists a sequence of successors (the same in every interpretation) that terminates in a state (the same in every interpretation) where G holds (in every interpretation).

[3] We use $\exists \mathbf{K}R_{M_i}.C$ as an abbreviation for $\exists \mathbf{K}R_{M_i}.\top \sqcap \forall \mathbf{K}R_{M_i}.C$. Indeed, since actions are assumed to be deterministic, the two concept expressions are equivalent.

4 Sensing

We now extend our framework in order to deal with sensing actions. Sensing actions are special actions that change the knowledge of the robot without changing the state of the external world in which the robot is embedded. Such an assumption may seem restrictive for sensing actions that can produce changes in the environment (i.e. sensing through the sonars may require motion). This is in fact not the case as we shall see later on, while it is useful to characterize sensing actions simply as knowledge producing actions.

We assume that the robot can sense certain facts represented by special atomic propositions, called *sensed propositions*; each sensed proposition is associated with a specific sensing action.

Action precondition axioms in Γ_P for the action R_S which senses the proposition S have the form:

$$\mathbf{K}C \sqcap \neg\mathbf{A}S \sqcap \neg\mathbf{A}\neg S \sqsubseteq \exists\mathbf{K}R_S.\top \tag{5}$$

where C is an $\mathcal{ALC}$ concept. It can be read as: if C holds in the current state s and the truth value of S is not known (i.e. it is consistent to assume both that S holds in s in every interpretation and that $\neg S$ holds in s in every interpretation), then it is possible to perform R_S, in the sense that there exists a unique R_S-successor s' of s which is the same in every interpretation.

Each sensing action R_S for the sensing proposition S has a *unique* **effect axiom** in Γ_E:

$$\mathbf{K}\top \sqsubseteq \forall\mathbf{K}R_S.\mathbf{K}S \sqcup \mathbf{K}\neg S \tag{6}$$

This axiom expresses that after having performed the action R_S the robot knows the truth value of the sensed proposition S, i.e. it knows *whether* S holds or not.

We enforce the following **frame axiom schemas** (Γ_{FR}):

$$\mathbf{K}\varphi \sqsubseteq \forall\mathbf{K}R_S.\mathbf{K}\varphi \tag{7}$$

one for each sensing action R_S, where φ stands for any $\mathcal{ALC}$ concept.[4] This propagates all concepts that hold in the current state s to the next state s'. The expression $\neg\mathbf{A}S \sqcap \neg\mathbf{A}\neg S$ in the premises of the precondition axioms for R_S prevents the execution of R_S in case either $\mathbf{K}S$ or $\mathbf{K}\neg S$ holds in the previous state. Hence, no contradiction may be generated from instances of the frame axiom schemas and the effect axiom for R_S.

[4] In fact, a number of instances which is linear in the size of the knowledge base suffices (see the following section).

Planning problem with sensing The robot's ability of sensing can be used to extend the notion of plan considered before to the notion of *conditional plan*. Indeed the robot may use its sensing capability to choose different courses of actions leading to a given goal, depending on the value of the sensed propositions. The planning problem then becomes:

$$\Sigma \models COND\_PLAN\_FOR\_G(init) \tag{8}$$

where: (i) Σ is the knowledge base including the static axioms Γ_S, the action precondition axioms Γ_P and the effect axioms Γ_E for both moving and sensing actions plus the frame axiom schema Γ_{FR} for the sensing actions, and the initial state description axioms Γ_I; (ii) $COND\_PLAN\_FOR\_G(init)$ denotes that $COND\_PLAN\_FOR\_G$ holds in the initial state, where $COND\_PLAN\_FOR\_G$ is *any* concept belonging to the set $\mathcal{P}_\mathcal{C}$ defined inductively as follows:

1. $\mathbf{K}G \in \mathcal{P}_\mathcal{C}$;
2. if $C \in \mathcal{P}_\mathcal{C}$, then $\exists \mathbf{K} R_{M_i}.C \in \mathcal{P}_\mathcal{C}$, for every moving action R_{M_i};
3. if $C_1, C_2 \in \mathcal{P}_\mathcal{C}$, then $\exists \mathbf{K} R_{S_i}.(\mathbf{K}S_i \sqcap C_1) \sqcup (\mathbf{K}\neg S_i \sqcap C_2) \in \mathcal{P}_\mathcal{C}$, for every sensing action R_{S_i}.

5 Plan generation

To the aim of generating plans in the framework proposed, we introduce the notion of *first-order extension* of a (epistemic) knowledge base $\Sigma = \Gamma_S \cup \Gamma_P \cup \Gamma_E \cup \Gamma_{FR} \cup \Gamma_I$ containing the specification of the robot's behavior in the terms described above. Informally, the first-order extension of Σ (denoted as $FOE(\Sigma)$) is an $\mathcal{ALC}$ knowledge base which consists of: (1) the static axioms in Γ_S; (2) the specification of the initial state (the assertions on *init* in Γ_I) augmented by the assertions which are consequences (up to renaming of individuals) of the epistemic sentences in Σ. The FOE of Σ provides a unique characterization of the knowledge that is shared by all the models of Σ, which is relevant wrt the planning problem.

In order to compute the first-order extension, we replace each sensing action R_S by two special actions R_S^+ and R_S^-. We denote by $\Gamma_E^\pm$ the set of effect axioms Γ_E in which those for the sensing actions R_S are replaced by:

$$\mathbf{K}\top \sqsubseteq \forall \mathbf{K} R_S^+.\mathbf{K}S \qquad\qquad \mathbf{K}\top \sqsubseteq \forall \mathbf{K} R_S^-.\mathbf{K}\neg S.$$

We also use only a finite number of instances of the frame axiom schemas. We denote by $\Gamma^{\pm}_{IFR}$ the set of axioms:

$$\mathbf{K}C \sqsubseteq \forall \mathbf{K}R^{+}_{S}.\mathbf{K}C \qquad \mathbf{K}C \sqsubseteq \forall \mathbf{K}R^{-}_{S}.\mathbf{K}C$$

obtained by: (1) instantiating the frame axiom schemas in Γ_{FR} for each concept C such that either $C(init) \in \Gamma_I$, or $\mathbf{K}C$ is in the postcondition of some effect axiom in Γ_E (i.e., C such that $\mathbf{K}D \sqsubseteq \forall \mathbf{K}R_M.\mathbf{K}C$, or $C, \neg C$ such that $\mathbf{K}\top \sqsubseteq \forall \mathbf{K}R_S.\mathbf{K}C \sqcup \mathbf{K}\neg C$ in Γ_E); (2) replacing each sensing action R_S by the two special actions R^{+}_{S} and R^{-}_{S}.

The FOE of Σ is computed by the following algorithm:

ALGORITHM FOE
INPUT: $\Sigma = \Gamma_S \cup \Gamma_P \cup \Gamma_E \cup \Gamma_{FR} \cup \Gamma_I$
OUTPUT: $FOE(\Sigma)$
  PROCEDURE CREATE_NEW_STATE(s, R)
  begin
    s' = NEW state name;
    $\mathcal{A}' = \mathcal{A} \cup \{R(s,s')\} \cup \{D(s') \mid D \in POST(s, R, \Gamma_S \cup \mathcal{A}, \Gamma^{\pm}_{E} \cup \Gamma^{\pm}_{IFR})\}$
    if there exists a state $s'' \in$ ALL_STATES **such that**
      $CONCEPTS(\Gamma_S \cup \mathcal{A}, s'') = CONCEPTS(\Gamma_S \cup \mathcal{A}', s')$
    then $\mathcal{A} = \mathcal{A} \cup R(s, s'')$
    else begin
        $\mathcal{A} = \mathcal{A}'$;
        ACTIVE_STATES = ACTIVE_STATES $\cup \{s'\}$;
        ALL_STATES = ALL_STATES $\cup \{s'\}$
      end
  end;
begin
  ACTIVE_STATES = $\{init\}$;
  ALL_STATES = $\{init\}$;
  $\mathcal{A} = \Gamma_I$;
  repeat
   s = choose(ACTIVE_STATES);
   for each moving action R_M **do**
    if there exists $\mathbf{K}C \sqsubseteq \exists \mathbf{K}R_M.\top \in \Gamma_P$
     such that $\Gamma_S \cup \mathcal{A} \models C(s)$
    then
     CREATE_NEW_STATE(s, R_M);
   for each sensing action R_S **do**
    if there exists $\mathbf{K}C \sqcap \mathbf{A}S \sqcap \mathbf{A}\neg S \sqsubseteq \exists \mathbf{K}R_S.\top \in \Gamma_P$
     such that $\Gamma_S \cup \mathcal{A} \models C(s)$ **and** $\Gamma_S \cup \mathcal{A} \not\models S(s)$ **and** $\Gamma_S \cup \mathcal{A} \not\models \neg S(s)$
    then begin

CREATE_NEW_STATE(s, R_S^+);
CREATE_NEW_STATE(s, R_S^-)
end;
ACTIVE_STATES = ACTIVE_STATES $-\{s\}$
until ACTIVE_STATES = $\emptyset$;
return $\Gamma_S \cup \mathcal{A}$
end.

In the above algorithm, $CONCEPTS(\Gamma_S \cup \mathcal{A}, s) = \{C \mid \Gamma_S \cup \mathcal{A} \models C(s)\}$ denotes the set of concepts that are *valid* for the explicitly named individual s, occurring in the set of instance assertions $\mathcal{A}$, wrt the $\mathcal{ALC}$ knowledge base $\Gamma_S \cup \mathcal{A}$. $POST(s, R, \Gamma_S \cup \mathcal{A}, \Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}) = \{D \mid \mathbf{K}C \sqsubseteq \forall \mathbf{K}R.\mathbf{K}D \in \Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}$ and $\Gamma_S \cup \mathcal{A} \models C(s)\}$ denotes the effect of the application of the all triggered rules belonging to the set $\Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}$ involving the action R in the state s, namely the set of postconditions (concepts) of the rules which are triggered by s.

Informally, the algorithm, starting from the initial state *init*, applies to each state the rules in the set $\Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}$ which are triggered by such a state. A new state is thus generated, unless a state with the same properties has already been created. In this way the effect of the rules is computed, obtaining a sort of "completion" of the knowledge base.

The FOE is unique, that is, every order of extraction of the states from the set $ACTIVE\_STATES$ produces the same set of assertions, up to renaming of states. Moreover, the algorithm terminates, that is, the condition $ACTIVE\_STATES = \emptyset$ is eventually reached, since the number of states generated is bound by the number of rules in $\Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}$. More precisely, the number of generated states n_s is $n_s \leq 2^{n_r+1}$ with n_r equal to the number of rules in $\Gamma_E^{\pm} \cup \Gamma_{IFR}^{\pm}$, i.e., $n_r = n_{em} + 2n_{es} + 2n_{fr}(n_e + n_i)$, where: n_{em} and n_{es} are the number of effect axioms in Γ_E for moving and sensing actions respectively (n_{es} is equal to the number of sensing actions); $n_{fr} = |\Gamma_{FR}|$ is the number of frame axiom schemas (which is again equal to the number of sensing actions); $n_e = |\Gamma_E|$ is the total number of effect axioms; $n_i = |\Gamma_I|$ is the number of initial state description axioms in Γ_I. Observe that n_s depends essentially on the size of Γ_E and Γ_I.

Finally, the condition $CONCEPTS(\Gamma_S \cup \mathcal{A}, s) = CONCEPTS(\Gamma_S \cup \mathcal{A}', s')$ can be checked by verifying whether, for each concept C such that either $C(init) \in \Gamma_I$ or $\mathbf{K}C$ is in the postcondition of some axiom in Γ_E, $\Gamma_S \cup \mathcal{A} \models C(s)$ iff $\Gamma_S \cup \mathcal{A}' \models C(s')$.

Next we show that the notion of first-order extension constitutes the basis of a sound and complete planning method. More specifically, we show that the planning problem in Σ expressed by (8) can be reduced to an entailment problem in $FOE(\Sigma)$, by making use of the following translation function $\tau(\cdot)$.

Definition Let C be a concept expression representing a plan (i.e. belonging to the set $\mathcal{P}_{\mathcal{C}}$). Then, $\tau(C)$ is the concept expression obtained as follows:

1. if $C = \mathbf{K}G$ then $\tau(C) = \mathbf{K}G$;
2. if $C = \exists \mathbf{K} R_{M_i}.C_1$ then $\tau(C) = \exists \mathbf{K} R_{M_i}.\tau(C_1)$;
3. if $C = \exists \mathbf{K} R_{S_i}.(\mathbf{K}S_i \sqcap C_1) \sqcup (\mathbf{K}\neg S_i \sqcap C_2)$ then $\tau(C) = \exists \mathbf{K} R^{+}_{S_i}.\tau(C_1) \sqcap \exists \mathbf{K} R^{-}_{S_i}.\tau(C_2)$.

□

Theorem 1 *Let $C \in \mathcal{P}_{\mathcal{C}}$. Then, $\Sigma \models C(init)$ iff $FOE(\Sigma) \models \tau(C)(init)$.*

6 Implementation

The framework previously presented has been actually used to describe the knowledge of the mobile robot Tino of the Erratic family [8]. In such implementation we use a restricted DL language to represent the robot's knowledge, which allows us to rely on the reasoning services provided by the well-known DL system CLASSIC [2]. In particular, we make use of the built-in instance checking mechanism to check the validity of a concept in a state, and of triggering of rules to propagate effects. However, CLASSIC does not provide an implementation for **K** and **A**, which are therefore handled by ad hoc attached procedures. The planning procedure, given an initial state and a goal, generates a conditional plan that, when executed starting from the initial state, leads to a state in which the goal is satisfied. Furthermore, dynamic execution of plans is supervised by the monitor, which is responsible for integrating planning and control.

Generating conditional plans Conditional plans can in principle be generated in two steps. First, the FOE of the knowledge base is generated (using the algorithm above); such a knowledge base can be seen as an action graph representing all possible plans starting from the initial state. Then, such a graph is visited, building a term (the conditional plan) representing a tree in which: (i) sensing actions generate branches; (ii) each branch leads to a state satisfying the goal. Obviously, several strategies can be applied to implement this method, and they are not addressed in this paper.

Dynamic execution of plans The reactive capabilities of the mobile robot Tino are based on a fuzzy controller [17, 7], which provides integrated routines for sonar sensor interpretation, map building, and navigation. The control problem is decomposed into small units of control, called (low-level) *behaviors*, that are distinguished in reactive ones like avoiding obstacles, and (low-level) goal-oriented ones, like following a corridor. A blending mechanism is used to integrate reactive and goal-oriented behaviors, so that the robot can follow a corridor while avoiding obstacles.

The integration of planning and control in our robot is characterized by the mapping between high-level actions and goal-oriented behaviors. Indeed, high-level actions can be seen as goals to be achieved by the controller through the activation of appropriate behaviors. The dynamic execution of plans is achieved by a monitor responsible of integrating planning and control by both translating high-level actions into goal-oriented behaviors and scheduling the activation of such behaviors, taking care of choosing the correct branch of the plan according to the result of sensing actions. The monitor also checks the correct execution of the behaviors handling the possible plan failures and requests for replanning.

The introduction of sensing actions requires the design and implementation of *sensing behaviors*. We have extended the set of behaviors of the robot by adding the *SenseOpenDoor* one, that is able to determine whether a specified door is open. This behavior is based on sonar sensor interpretation and returns to the monitor a truth value, indicating the success of a sensing action, which will be used to decide which action to perform next. Notably, the robot moves while performing this behavior. However, the state is not specified by the absolute position of the robot, but rather by a property, such as being close to the door. The action *SenseOpenDoor* does not change the state of the robot since it is expected to leave the robot in the state close to the door, even if the actual position may have changed. Such a change does not matter as long as the subsequent action (for example entering the door) can be successfully accomplished.

7 Conclusions

In this paper we have proposed a logical framework for reasoning about action which provides for the formalization of sensing actions. In particular, we have shown that the use of the epistemic state of the agent (represented through the modal operators of **K** and **A**) allows for the formalization of sensing actions. Our approach has strong connections with previous research on logical formalization of knowledge-producing actions [18, 13], which has pointed out the fact that a formalization of sensing actions must satisfy the properties of *non-forgetting* and *minimal learning*.

Our work on sensing is also related to [6], which presents a formalization of sensing actions based on an extension of STRIPS constructs. Sensing actions are distinguished by means of an annotation mechanism on the post-conditions. A particular use of annotated propositions allows for expressing constraints on the plan, for example sensing the color of a door in order to enter into a room with a blue door is allowed, while painting a door blue in order to enter into such a room is forbidden. We are currently studying the possibility of adding plan constraints to our planner. In particular, we want to exploit the ideas reported in [1], which have been shown effective for speeding up the planning process.

References

[1] F. Bacchus and F. Kabanza. Using temporal logic to control search in a forward chaining planner. In *Proc. of the 3rd European Workshop on Planning*, 1995.

[2] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. of ACM SIGMOD*, pages 59–67, 1989.

[3] G. De Giacomo, L. Iocchi, D. Nardi, and R. Rosati. Moving a robot: the KR&R approach at work. In *Proc. of KR-96*, 1996.

[4] G. De Giacomo and M. Lenzerini. Boosting the correspondence between description logics and propositional dynamic logics. In *Proc. of AAAI-94*, pages 205–212, 1994.

[5] F. M. Donini, D. Nardi, and R. Rosati. Autoepistemic description logics. In *Proc. of IJCAI-97*, pages 136–141, 1997.

[6] O. Etzioni, S. Hanks, D. Weld, D. Draper, N. Lesh, and M. Williamson. An approach to planning with incomplete information. *Proc. of KR-92*, 1992.

[7] K. Konolige, K. Myers, E. Ruspini, and A. Saffiotti. The Saphira architecture: A design for autonomy. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(1):215–235, 1997.

[8] K. Konolige. Erratic competes with the big boys. *AAAI Magazine*, 2: 61–67, 1995.

[9] D. Kozen and J. Tiuryn. Logics of programs. In *Handbook of Theoretical Computer Science*, pages 790–840. Elsevier, 1990.

[10] Y. Lesperance, H. J. Levesque, F. Lin, D. Marcu, R. Reiter, and R. B. Scherl. A logical approach to high-level robot programming. In *AAAI Fall Symposium on Control of the Physical World by Intelligent Systems*, 1994.

[11] H. J. Levesque. What is planning in presence of sensing? In *Proc. of AAAI-96*, pages 1139–1149, 1996.

[12] V. Lifschitz. Minimal belief and negation as failure. *AIJ*, 70:53–72, 1994.

[13] F. Lin and Y. Shoham. On non-forgetting and minimal learning. In *Proc. of the 1993 Int. Coll. on Cognitive Science*, 1993.

[14] R. Reiter. *Representation and Reasoning for Dynamic Systems*, 1997. Forthcoming.

[15] R. Reiter. A logic for default reasoning. *AIJ*, 13:81–132, 1980.

[16] S. Rosenschein. Plan synthesis: a logical approach. In *Proc. of IJCAI-81*, 1981.

[17] A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *AIJ*, 76(1-2):481–526, 1995.

[18] R. Scherl and H. J. Levesque. The frame problem and knowledge producing action. In *Proc. of AAAI-93*, pages 689–695, 1993.

[19] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of IJCAI-91*, pages 466–471, 1991.

[20] M. Shanahan. *Solving the Frame Problem: A Mathematical Investigation of the Common Sense Law of Inertia.* The MIT Press, 1997.