Semantics for a useful fragment of the situation calculus

Gerhard Lakemeyer
Dept. of Computer Science
RWTH Aachen
52056 Aachen
Germany
gerhard@cs.rwth-aachen.de

Hector J. Levesque
Dept. of Computer Science
University of Toronto
Toronto, Ontario
Canada M5S 3A6
hector@cs.toronto.edu

Abstract

In a recent paper, we presented a new logic called $\mathcal{ES}$ for reasoning about the knowledge, action, and perception of an agent. Although formulated using modal operators, we argued that the language was in fact a dialect of the situation calculus but with the situation terms suppressed. This, we claim, allows us to develop a clean and workable semantics for the language without piggybacking on the generic Tarski semantics for first-order logic. In this paper, we reconsider the relation between $\mathcal{ES}$ and the situation calculus and show how to map sentences of $\mathcal{ES}$ into the situation calculus. We argue that the fragment of the situation calculus represented by $\mathcal{ES}$ is rich enough to handle the basic action theories defined by Reiter as well as Golog. Finally, we show that in the full second-order version of $\mathcal{ES}$, almost all of the situation calculus can be accommodated.

1 Introduction

In a recent paper, we [2004] presented a new logic called $\mathcal{ES}$ for reasoning about the knowledge, action, and perception of an agent. Our main justification for introducing yet another knowledge representation formalism was twofold. First, we claimed that the language was not a new formalism at all, but merely a situation-suppressed version of part of the situation calculus, as originally presented by McCarthy and Hayes [McCarthy and Hayes 1969] and subsequently formalized by Reiter [Reiter 2001]. Second, we claimed that because the language was defined semantically rather than axiomatically, certain mathematical arguments about the formalism were considerably simpler than in the original situation calculus. We presented evidence for this by showing a very compact proof of the correctness of Reiter's regression operator, and another involving a property called the determinacy of knowledge.

However, our first claim, which was that $\mathcal{ES}$ is in fact just an alternate way of writing some formulas of the situation calculus, was left unsubstantiated. In this paper, we remedy this. The main result we prove is the correctness of a simple mapping between sentences of $\mathcal{ES}$ and their counterparts in the classical language of the situation calculus. In addition, we show that the fragment of the situation calculus expressed by $\mathcal{ES}$ is rich enough to handle (among other things) basic action theories (as defined by Reiter) and the *Do* operator which is the basis of the Golog language [Levesque et al. 1997]. We also illustrate this richness more informally using an example involving the knowledge, action, and sensing of a simple robot. However, there are sentences of the situation calculus that cannot be expressed directly in $\mathcal{ES}$, and which may be useful in some contexts. In the final part of the paper, we show that if we are prepared to use second-order quantification, there is a way to encode (almost) every sentence of the situation calculus in $\mathcal{ES}$. This encoding is a bit of a trick, and not something we advocate for everyday use. But it does exist. So we gain the clarity that comes from having a real semantic basis, and the ability to present concise semantic arguments, without any significant loss of expressiveness.

The rest of the paper is organized as follows. In the next section, we define the full language of $\mathcal{ES}$, its syntax and semantics. This generalizes the version of $\mathcal{ES}$ presented in [Lakemeyer and Levesque, 2004] in a variety of ways, the most important of which is that it allows functions, predicates, and second-order variables that are *fluent* (vary from situation to situation) as well as *rigid* (fixed for all situations). In Section 3, we consider the mapping from $\mathcal{ES}$ to the situation calculus and prove that a sentence of $\mathcal{ES}$ is valid iff its mapping is a logical entailment of a suitable situation calculus theory. In Section 4, we consider the expressiveness of $\mathcal{ES}$, first with an example basic action theory and then with *Do*. In Section 5, we consider the inverse mapping from the situation calculus to $\mathcal{ES}$, followed by some remarks on related work and conclusions. Then we stop.

2 The language: syntax and semantics

The full language $\mathcal{ES}$ cconsists of formulas over symbols from the following vocabulary:

- first-order variables: $x_1, x_2, \ldots, y_1, y_2, \ldots, a_1, a_2, \ldots$;
- fluent second-order variables of arity k: $P_1^k, P_2^k, \ldots$;
- rigid second-order variables of arity k: $Q_1^k, Q_2^k, \ldots$;
- standard names: $n_1, n_2, \ldots$;
- fluent function symbols of arity k: $f_1^k, f_2^k, \ldots$;
  for example, *location*, *bestAction*;

- rigid function symbols of arity k: $g_1^k, g_2^k, \ldots$; for example, *block5*, *repair*;
- fluent predicate symbols of arity k: $F_1^k, F_2^k, \ldots$; for example, *Broken*;[1]
- rigid predicate symbols of arity k: $G_1^k, G_2^k, \ldots$; for example, *Fragile*;
- connectives and other symbols: $=$, $\wedge$, $\neg$, $\forall$, *Know*, $\Box$, round and square parentheses, period, comma.

We assume that first-order variables, standard names, and function symbols come in two sorts, *action* (like *repair* and *bestAction*) and *object* (like *block5* and *location*). Constants are function symbols of 0 arity.[2] We let $\mathcal{N}$ denote the set of all standard names and $\mathcal{Z}$ denote the set of all sequences of standard names for actions, including $\langle\,\rangle$, the empty sequence.

2.1 Terms and formulas

The *terms* of the language are of sort *action* or *object*, and form the least set of expressions such that

1. Every standard name and first-order variable is a term of the corresponding sort;
2. If $t_1, \ldots, t_k$ are terms and h is a k-ary function symbol then $h(t_1, \ldots, t_k)$ is a term of the same sort as h.

By a *primitive term* we mean one of the form $h(n_1, \ldots, n_k)$ where h is a (fluent or rigid) function symbol and all of the n_i are standard names.

The *well-formed formulas* of the language form the least set such that

1. If $t_1, \ldots, t_k$ are terms, and H is a k-ary predicate symbol then $H(t_1, \ldots, t_k)$ is an (atomic) formula;
2. If $t_1, \ldots, t_k$ are terms, and V is a k-ary second-order variable, then $V(t_1, \ldots, t_k)$ is an (atomic) formula;
3. If t_1 and t_2 are terms, then $(t_1 = t_2)$ is a formula;
4. If t is an action term and α is a formula, then $[t]\alpha$ is a formula;
5. If α and β are formulas, v is a first-order variable, and V is a second-order variable, then the following are also formulas: $(\alpha \wedge \beta)$, $\neg\alpha$, $\forall v.\, \alpha$, $\forall V.\, \alpha$, $\Box\alpha$, *Know*(α).

We read $[t]\alpha$ as "α holds after action t", and $\Box\alpha$ as "α holds after any sequence of actions." As usual, we treat $(\alpha \vee \beta)$, $(\alpha \supset \beta)$, $(\alpha \equiv \beta)$, $\exists v.\, \alpha$, and $\exists V.\, \alpha$ as abbreviations. We call a formula without free variables a *sentence*. By a *primitive sentence* we mean a formula of the form $H(n_1, \ldots, n_k)$ where H is a (fluent or rigid) predicate symbol and all of the n_i are standard names.

2.2 The semantics

The language contains both fluent and rigid expressions. The former vary as the result of actions and have values that may be unknown, but the latter do not. Intuitively, to determine whether or not a sentence α is true after a sequence of actions z has been performed, we need to specify two things: a world w and and an epistemic state e. We write $e, w, z \models \alpha$. A world determines truth values for the primitive sentences and coreferring standard names for the primitive terms after any sequence of actions. An epistemic state is defined by a set of worlds, as in possible-world semantics. More precisely:

- a world $w \in W$ is any function from the primitive sentences and $\mathcal{Z}$ to $\{0, 1\}$, and from the primitive terms and $\mathcal{Z}$ to $\mathcal{N}$ (preserving sorts), and satisfying the rigidity constraint: if r is a rigid function or predicate symbol, then $w[r(n_1, \ldots, n_k), z] = w[r(n_1, \ldots, n_k), z']$, for all z and z' in $\mathcal{Z}$.
- an epistemic state $e \subseteq W$ is any set of worlds.

We extend the idea of coreferring standard names to arbitrary ground terms as follows. Given a term t without variables, a world w, and an action sequence z, we define $|t|_w^z$ (read: the coreferring standard name for t given w and z) by:

1. If $t \in \mathcal{N}$, then $|t|_w^z = t$;
2. $|h(t_1, \ldots, t_k)|_w^z = w[h(n_1, \ldots, n_k), z]$, where $n_i = |t_i|_w^z$.

To interpret formulas with free variables, we proceed as follows. First-order variables are handled substitutionally using the standard names. To handle the quantification over second-order variables, we use second-order *variable maps* defined as follows:

> The *second-order primitives* are formulas of the form $V(n_1, \ldots, n_k)$ where where V is a (fluent or rigid) second-order variable and all of the n_i are standard names. A *variable map* u is a function from worlds, second-order primitives, and $\mathcal{Z}$ to $\{0, 1\}$, satisfying the rigidity constraint: if Q is a rigid second-order variable, then for all w and w' in W, and all z and z' in $\mathcal{Z}$, $u[w, Q(n_1, \ldots, n_k), z] = u[w', Q(n_1, \ldots, n_k), z']$.

Let u and u' be variable maps, and let V be a (fluent or rigid) second-order variable; we write $u' \sim_V u$ to mean that u and u' agree except perhaps on the second-order primitives involving V. Finally, to interpret what is known after a sequence of actions possibly including sensing has taken place, we define $w' \simeq_z w$ (read: w' and w agree on the sensing throughout action sequence z) inductively by the following:

1. $w' \simeq_{\langle\,\rangle} w$ iff w' and w agree on the value of every primitive rigid term and sentence;
2. $w' \simeq_{z \cdot n} w$ iff $w' \simeq_z w$ and $w'[SF(n), z] = w[SF(n), z]$.

Putting all these together, here is the semantic definition of truth. Given $e \subseteq W$ and $w \in W$, we define $e, w \models \alpha$ (read: α is true) as $e, w, \langle\,\rangle \models \alpha$, where for any $z \in \mathcal{Z}$ and any second-order variable map u:

1. $e, w, z, u \models H(t_1, \ldots, t_k)$ iff $w[H(n_1, \ldots, n_k), z] = 1$, where $n_i = |t_i|_w^z$;
2. $e, w, z, u \models V(t_1, \ldots, t_k)$ iff $u[w, V(n_1, \ldots, n_k), z] = 1$, where $n_i = |t_i|_w^z$;

[1] We assume this list includes the predicates *Poss* and *SF*.

[2] The standard names can be thought of as special extra constants that satisfy the unique name assumption and an infinitary version of domain closure.

3. $e, w, z, u \models (t_1 = t_2)$ iff
n_1 and n_2 are identical, where $n_i = |t_i|^z_w$;
4. $e, w, z, u \models [t]\alpha$ iff $e, w, z{\cdot}n, u \models \alpha$, where $n = |t|^z_w$;
5. $e, w, z, u \models (\alpha \wedge \beta)$ iff
$e, w, z, u \models \alpha$ and $e, w, z, u \models \beta$;
6. $e, w, z, u \models \neg\alpha$ iff $e, w, z, u \not\models \alpha$;
7. $e, w, z, u \models \forall v.\, \alpha$ iff $e, w, z, u \models \alpha^x_n$,
for every standard name n (of the same sort as v);
8. $e, w, z, u \models \forall V.\, \alpha$ iff
$e, w, z, u' \models \alpha$, for every $u' \sim_V u$;
9. $e, w, z, u \models \Box\alpha$ iff
$e, w, z \cdot z', u \models \alpha$, for every $z' \in \mathcal{Z}$;
10. $e, w, z, u \models \mathit{Know}(\alpha)$ iff
$e, w', z, u \models \alpha$, for every $w' \in e$ such that $w' \simeq_z w$.

When α is *objective* (has no *Know* operators), we can leave out the e and write $w \models \alpha$. When Σ is a set of sentences and α is a sentence, we write $\Sigma \models \alpha$ (read: Σ logically entails α) to mean that for every e and w, if $e, w \models \alpha'$ for every $\alpha' \in \Sigma$, then $e, w \models \alpha$. Finally, we write $\models \alpha$ (read: α is valid) to mean $\{\} \models \alpha$.

3 Mapping to the situation calculus

How do we know that the semantics of $\mathcal{ES}$ is correct? In this section, we argue that it is indeed correct by showing how formulas α in $\mathcal{ES}$ can be translated in a direct way to formulas α^* in the situation calculus as defined by Reiter.[3]

The most desirable and simplest outcome of this translation would be that $\models \alpha$ iff $\Sigma \models_{\mathrm{FOL}} \alpha^*$, where $\models$ is validity in $\mathcal{ES}$, Σ is the set of foundational axioms of the situation calculus (see [Levesque *et al*, 1998], for example), and $\models_{\mathrm{FOL}}$ is ordinary classical logical consequence. Unfortunately, we do not get exactly this correspondence for a variety of reasons we will discuss below. But we do get something close:

$$\models \alpha \quad \textit{iff} \quad \Sigma \cup \Upsilon \models_{\mathrm{FOL}} \alpha^*,$$

where Υ is a set of four axioms that we will justify separately.

To prove this result it will be necessary to work with ordinary Tarski models of sentences of the situation calculus. As argued in [Lakemeyer and Levesque, 2004], this is difficult and painstaking, and is indeed one of the main reasons to prefer $\mathcal{ES}$ over the situation calculus. So while the proof of the theorem is quite laborious, we remind the reader that this can be thought of as a final reckoning for a formalism that is unworkable semantically. For space reasons, we do not review the conventional notation used for talking about Tarski interpretations, denotations, extensions, variable maps. See, for example, [Enderton 1972].

[3]We assume that this language has functional and relational fluents, functions and predicates that are not fluents, the distinguished constant S_0, function *do*, predicates $\sqsubset$, *Poss* and *SF*, and a two-place predicate K for knowledge. We take $\mathit{Knows}(\alpha, \sigma)$ in the situation calculus as an abbreviation for the formula $\forall s(K(s, \sigma) \supset \alpha^{\mathit{now}}_s)$, where α^{now}_s is the result of replacing by s in α every occurrence of *now* that is not within the scope of a further *Know*s.

3.1 Definition of the translation

Before describing Υ, we present the translation from $\mathcal{ES}$ into the situation calculus. In the simplest case, the idea is that a formula like $\mathit{Broken}(c)$, where *Broken* is a fluent, will be mapped to the situation calculus formula $\mathit{Broken}(c, S_0)$, where we have restored the distinguished situation term S_0 for the fluent. Similarly, $[\mathit{repair}(c)]\neg\mathit{Broken}(c)$ will be mapped to $\neg\mathit{Broken}(c, \mathit{do}(\mathit{repair}(c), S_0))$. So $\mathcal{ES}$ formulas can be thought of "situation-suppressed" (in situation-calculus terminology) and the * mapping restores the situation argument to the fluents, leaving the rigids unchanged.

More precisely, we have the following:

Definition 3.1 *Let α be any term or formula of $\mathcal{ES}$ without standard names. The expression α^* is defined as $\alpha[S_0]$ where, for any situation term σ, $\alpha[\sigma]$ is defined inductively by:*

1. *$v[\sigma]$, where v is a first-order variable, is v;*
2. *$g(t_1, \ldots, t_k)[\sigma]$, where g is a rigid function, predicate, or second-order variable, is $g(t_1[\sigma], \ldots, t_k[\sigma])$;*
3. *$f(t_1, \ldots, t_k)[\sigma]$, where f is a fluent function, predicate, or second-order variable is $f(t_1[\sigma], \ldots, t_k[\sigma], \sigma)$;*
4. *$(t_1 = t_2)[\sigma]$ is $(t_1[\sigma] = t_2[\sigma])$;*
5. *$([t]\alpha)[\sigma]$ is $\alpha[\mathit{do}(t[\sigma], \sigma)]$;*
6. *$(\alpha \wedge \beta)[\sigma]$ is $(\alpha[\sigma] \wedge \beta[\sigma])$*
7. *$(\neg\alpha)[\sigma]$ is $\neg\alpha[\sigma]$;*
8. *$(\forall v.\, \alpha)[\sigma]$ is $\forall v.\, \alpha[\sigma]$;*
9. *$(\forall V.\, \alpha)[\sigma]$ is $\forall V.\, \alpha[\sigma]$.*
10. *$(\Box\alpha)[\sigma]$ is $\forall s'(\sigma \sqsubseteq s' \supset \alpha[s'])$;*
11. *$\mathit{Know}(\alpha)[\sigma]$ is $\mathit{Knows}(\alpha[\mathit{now}], \sigma)$.*

Note that the translation of $\Box\alpha$ introduces quantification over situations, where the introduced variable s' is assumed to be one that does not appear in situation term σ.

3.2 The situation-calculus axioms Υ

The axioms we assume in Υ are the following:

1. domain of objects is countably infinite;[4]
2. domain of actions is countably infinite (as above);
3. equality is the identity relation:
$\forall x \forall y.\, (x = y) \equiv \forall Q(Q(x) \equiv Q(y))$.
4. the K predicate:[5] $\forall s(K(s, S_0) \supset \mathit{Ini}(s)) \wedge$
$\forall s' \forall s.\, K(s', s) \equiv \forall P(\ldots \supset P(s', s))$,
where the ellipsis stands for the universal closure of
$[K(s_1, S_0) \wedge \mathit{Ini}(s_2) \supset P(s_1, s_2)] \wedge$
$[P(s_1, s_2) \wedge \mathit{SF}(a, s_1) \equiv \mathit{SF}(a, s_2) \supset$
$P(\mathit{do}(a, s_1), \mathit{do}(a, s_2))]$.

[4]For space reasons, we omit the formula of second-order logic that fixes the cardinality of the domain of objects and actions.

[5]We let $\mathit{Ini}(t)$ be an abbreviation for the situation calculus formula $\forall a \forall s(t \neq \mathit{do}(a, s))$. In this version of the axiom, we ignore the correspondence between K and *Poss*.

Axioms (1) and (2) talk about the cardinality of the set of objects and actions respectively: they are both countable and infinite. The countability aspect is not very controversial. In the first-order case, every satisfiable set of sentences is satisfiable in a countable domain, and we do not expect users of the situation calculus to use second-order logic to defeat this. We can, however, imagine contexts where finiteness might be desirable. In such cases, we can introduce a new predicate O and instead of asserting that there are finitely many objects, assert that there are finitely many objects in O.

As for axiom (3), it is hard imagining anyone taking the negation of this one seriously. The usual first-order axiomatization of equality is often enough, but the intent is invariably for the equality symbol to be understood as the identity relation, which this second-order axiom ensures.

Finally axiom (4) is a second order definition of the K predicate in terms of the value it has at S_0. This is just another way of capturing the successor state axiom for K introduced by Scherl and Levesque [2003], and the added machinery to make *Knows* be a weak-S5 operator [Hughes and Cresswell, 1968]. Other knowledge operators are possible in the situation calculus, but weak-S5 and its extensions (such as strong-S5) are the most often used.

3.3 The embedding theorem

What we show is that provided certain properties hold between a Tarski structure on one side and an epistemic state, and a collection of worlds on the other, sentences will be true in $\mathcal{ES}$ iff their translations are true in the Tarski structure.

Let $\mathcal{M}$ be a Tarski structure for the situation calculus over the domain $\mathcal{D} = \mathcal{D}_{sit} \cup \mathcal{D}_{act} \cup \mathcal{D}_{obj}$, with $\mathcal{D}_{ini} \subseteq \mathcal{D}_{sit}$ as the set of initial situations, and with $\iota_0 = S_0^{\mathcal{M}} \in \mathcal{D}_{ini}$. Let e be an epistemic state, and assume that we are given three mappings $\omega \in [\mathcal{D}_{ini} \rightarrow W]$, $\theta \in [\mathcal{N} \rightarrow \mathcal{D}_{act} \cup \mathcal{D}_{obj}]$, and $\pi \in [\mathcal{Z} \times \mathcal{D}_{ini} \rightarrow \mathcal{D}_{sit}]$. These mappings need to satisfy various consistency properties that we cannot enumerate here. But we can at least quote the lemma that uses them:

Lemma 3.2 *Let α be a formula of $\mathcal{ES}$ with no standard names and whose free variables are among $x_1, \ldots, x_m$. Then, given the consistency properties, for any variable map μ, the $\mathcal{ES}$ variable map u_μ (defined in terms of μ), any variable s, any $z \in \mathcal{Z}$, any $\iota \in \mathcal{D}_{ini}$, and $w = \omega(\iota)$,*

$$e, w, z, u_\mu \models \alpha^{x_1 \ldots x_m}_{n_1 \ldots n_m}$$
$$\textit{iff}$$
$$M, \mu\{x_1/\theta(n_1), \ldots, x_m/\theta(n_m), s/\pi(z, \iota)\} \models \alpha[s].$$

This is proved by a (long) induction on the structure of α. The main result is then the following:

Theorem 3.3 *Let α be any sentence of $\mathcal{ES}$ without standard names. Then α is valid iff $\Sigma \cup \Upsilon \models_{\rm FOL} \alpha^*$.*

Proof: (Sketch) First assume that α is not valid. Then there is an e, w_0 such that $e, w_0 \not\models \alpha$. We define a Tarski structure $\mathcal{M}$ whose domain is $\mathcal{D} = \mathcal{D}_{obj} \cup \mathcal{D}_{act} \cup \mathcal{D}_{sit}$, where $\mathcal{D}_{obj}$ (resp. $\mathcal{D}_{act}$) is the set of standard names of objects (resp. actions), and $\mathcal{D}_{sit} = \mathcal{Z} \times W$. Then we define $=^{\mathcal{M}}$, $S_0^{\mathcal{M}}$, $do^{\mathcal{M}}$, $\sqsubset^{\mathcal{M}}$, $K^{\mathcal{M}}$, as well as the extension of every rigid and fluent predicate and function symbol using e and w_0 in a way that ensures that $\mathcal{M}$ satisfies $\Sigma \cup \Upsilon$. Next, define the mappings θ, π, and ω by letting $\theta(n) = n$, and for any initial $\iota = (\langle\rangle, w)$, letting $\pi(z, \iota) = (z, w)$, and $\omega(\iota) = w$. This ensures that the properties needed for Lemma 3.2 are satisfied, and therefore $\mathcal{M} \not\models \alpha^*$. Consequently, $\Sigma \cup \Upsilon \not\models_{\rm FOL} \alpha^*$.

Conversely, assume that $\Sigma \cup \Upsilon \not\models_{\rm FOL} \alpha^*$. Then there is a Tarski structure $\mathcal{M}$ that satisfies $\Sigma \cup \Upsilon$ but such that $\mathcal{M} \not\models \alpha^*$. The domain $\mathcal{D}$ must be $\mathcal{D}_{sit} \cup \mathcal{D}_{act} \cup \mathcal{D}_{obj}$, with $\mathcal{D}_{ini} \subseteq \mathcal{D}_{sit}$ as the set of initial situations, and with $\iota_0 = S_0^{\mathcal{M}} \in \mathcal{D}_{ini}$. Since $\mathcal{M} \models \Upsilon$, both $\mathcal{D}_{obj}$ and $\mathcal{D}_{act}$ are countably infinite, $\mathcal{D}_{obj} = \{\delta_1, \delta_2, \ldots\}$, and $\mathcal{D}_{act} = \{\lambda_1, \lambda_2, \ldots\}$. We define θ to map the i-th standard name for objects to δ_i and analogously for actions. We then define π using $do^{\mathcal{M}}$, and ω using the extensions of the function and predicate symbols given by $\mathcal{M}$. Finally, we let $e = \{\omega(\iota) \mid (\iota, \iota_0) \in K^{\mathcal{M}}\}$, and $w_0 = \omega(\iota_0)$. These ensure that the properties needed for Lemma 3.2 are satisfied, and so $e, w_0 \not\models \alpha$. Consequently, α is not valid. ∎

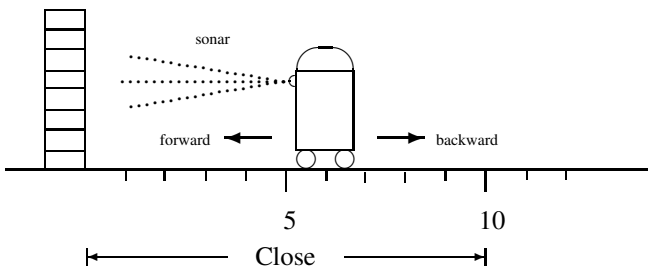

Figure 1: A simple robot

4 The expressiveness of the language

Now that we have established that $\mathcal{ES}$ is actually a disguised fragment of the situation calculus, we next consider the expressiveness of this fragment, starting with a simple example problem, adapted from [Levesque and Lakemeyer 2001].

Imagine a robot that lives in a 1-dimensional world, and that can move towards or away from a fixed wall. The robot also has a sonar sensor that tells it when it gets close to the wall, say, less than 10 units away. See Figure 1. So we might imagine three actions, *forward* and *backward* which move the robot one unit towards and away from the wall, and a *sonar* sensing action which tells the robot if it is close to the wall. Each of these is a rigid constant, but for simplicity, we will simply assume that they are standard names. We have a single fluent, *distance*, which gives the actual distance from the robot to the wall. We can use *Close* as an abbreviation for the formula "*distance* < 10." [6] We begin our formalization by writing preconditions for the three actions:

$$\begin{aligned}\forall a \,\Box\, \textit{Poss}(a) \;\equiv\; & \\ & a = \textit{forward} \;\wedge\; \textit{distance} > 0 \quad \vee \\ & a = \textit{backward} \;\wedge\; \text{TRUE} \quad \vee \\ & a = \textit{sonar} \;\wedge\; \text{TRUE}.\end{aligned}$$

Next, we define the sensing results for the actions:

[6] Here and below, we use simple arithmetic involving $<$, $+$, and $-$, which can easily be defined in second-order terms with the standard names acting as natural numbers. We omit the details.

$\forall a \,\Box\, \mathit{SF}(a) \;\equiv$
$\quad a = \mathit{forward} \;\wedge\; \text{TRUE} \quad \vee$
$\quad a = \mathit{backward} \;\wedge\; \text{TRUE} \quad \vee$
$\quad a = \mathit{sonar} \;\wedge\; \mathit{Close}.$

Finally, we write a successor state axiom for our only fluent:

$\forall a, x \,\Box\, [a](\mathit{distance} = x) \;\equiv$
$\quad a = \mathit{forward} \;\wedge\; \mathit{distance} = x + 1 \quad \vee$
$\quad a = \mathit{backward} \;\wedge\; \mathit{distance} = x - 1 \quad \vee$
$\quad a \neq \mathit{forward} \;\wedge\; a \neq \mathit{backward} \;\wedge\; \mathit{distance} = x.$

Now we are ready to consider some specifics having to do with what is true initially by defining an action theory Σ. Let ϕ denote the conjunction of the sentences above. We assume that ϕ is true and the robot knows it. We also assume the robot is located initially 6 units away from the wall, but that the robot has no idea where it is. So, we let Σ be

$$\{\phi,\ \mathit{Know}(\phi),\ \mathit{distance} = 6,\ \forall x \neg \mathit{Know}(\mathit{distance} \neq x)\},$$

and we get this:

Theorem 4.1 *The following are logical entailments of* Σ:

1. $\mathit{Close} \wedge \neg \mathit{Know}(\mathit{Close}) \wedge [\mathit{forward}]\, \neg \mathit{Know}(\mathit{Close})$
   the robot is close to the wall, but does not know it, and continues not to know it after moving forward;
2. $[\mathit{sonar}]\, (\mathit{Know}(\mathit{Close}) \wedge [\mathit{forward}] \mathit{Know}(\mathit{Close}))$
   after reading the sonar, the robot knows it is close, and continues to know it after moving forward;
3. $[\mathit{sonar}]\, [\mathit{backward}]\, \neg \mathit{Know}(\mathit{Close})$
   after reading the sonar and then moving backward, the robot no longer knows that it is close to the wall;
4. $[\mathit{backward}]\, [\mathit{sonar}]\, \mathit{Know}(\mathit{Close})$
   after moving backward and then reading the sonar, the robot knows that it is close to the wall;
5. $[\mathit{sonar}]\, [\mathit{forward}]\, [\mathit{backward}]\, \mathit{Know}(\mathit{Close})$
   after reading the sonar, moving forward, and then backward, the robot knows that it is still close to the wall;
6. $[\mathit{sonar}]\, \mathit{Know}([\mathit{forward}]\, \mathit{Close})$
   after reading the sonar, the robot knows that it will remain close after moving forward;
7. $\neg \mathit{Know}([\mathit{sonar}]\, \mathit{Know}(\mathit{Close}))$
   the robot does not know initially that it will know that it is close after reading the sonar;
8. $\mathit{Know}([\mathit{sonar}]\, (\mathit{Know}(\mathit{Close}) \vee \mathit{Know}(\neg \mathit{Close})))$
   the robot does know initially that after reading the sonar, it will then know whether or not it is close to the wall;
9. $\mathit{Know}([\mathit{sonar}]\, [\mathit{backward}]\, \neg \mathit{Know}(\mathit{Close}))$
   the robot knows initially that it will not know that it is close after reading the sonar and moving backwards.

Proof: The proofs of these are similar. Here we will only do item 3. Let $z = \langle \mathit{sonar} \cdot \mathit{backward} \rangle$, and suppose that $e, w \models \Sigma$; we must show that $e, w, z \models \neg \mathit{Know}(\mathit{Close})$. Because $e, w \models \neg \mathit{Know}(\mathit{distance} \neq 9)$, there exists $w' \in e$ such that $w' \simeq_{\langle\rangle} w$ and $w'[\mathit{distance}, \langle\rangle] = 9$. Since $9 < 10$, we also have that $w' \simeq_z w$. However, $w'[\mathit{distance}, z] = 10$. So there exists $w' \in e$ such that $w' \simeq_z w$ and $w', z \models \neg \mathit{Close}$. Therefore, $e, w, z \models \neg \mathit{Know}(\mathit{Close})$. ∎

Although we will not attempt to formulate a theorem here, it should be clear from this example that any basic action theory [Reiter 2001] including those involving the Scherl-Levesque knowledge operator can be expressed in $\mathcal{ES}$.

4.1 The *Do* operator

We now turn our attention to the *Do* operator which is the basis of the Golog language [Levesque et al. 1997], and show that a variant of *Do* can be represented in $\mathcal{ES}$. We cannot encode $\mathit{Do}(\delta, s, s')$ directly since we do not have situations as terms. Instead we will use $\mathit{D\tilde{o}}(\delta, \alpha)$ which intuitively means that α holds after doing Golog program δ. There are two possible readings: one says that α holds in *all* final states, and the other says that α holds in *some* final state. They can be interdefined, so we consider only the latter. We treat $\mathit{D\tilde{o}}(\delta, \alpha)$ as an abbreviation for a formula of $\mathcal{ES}$, defined recursively on the δ as follows:

1. for any action a, $\mathit{D\tilde{o}}(a, \alpha) \stackrel{def}{=} (\mathit{Poss}(a) \wedge [a]\alpha)$;
2. $\mathit{D\tilde{o}}(\phi?,\ \alpha) \stackrel{def}{=} (\phi \wedge \alpha)$;
3. $\mathit{D\tilde{o}}(\delta\,;\,\delta',\ \alpha) \stackrel{def}{=} \mathit{D\tilde{o}}(\delta,\ \mathit{D\tilde{o}}(\delta', \alpha))$;
4. $\mathit{D\tilde{o}}(\delta \mid \delta',\ \alpha) \stackrel{def}{=} (\mathit{D\tilde{o}}(\delta, \alpha) \vee \mathit{D\tilde{o}}(\delta', \alpha))$;
5. $\mathit{D\tilde{o}}(\pi x.\ \delta,\ \alpha) \stackrel{def}{=} \exists x.\ \mathit{D\tilde{o}}(\delta, \alpha)$;
6. $\mathit{D\tilde{o}}(\delta^*,\ \alpha) \stackrel{def}{=}$
   $\forall P.\ \{\Box(\alpha \supset P) \wedge \Box(\mathit{D\tilde{o}}(\delta, P) \supset P)\} \supset P.$

As usual, we can define while-loops and if-then-else as abbreviations. The main theorem that we state here is that this account of $\mathit{D\tilde{o}}$ is correct relative to the original account of *Do* and the mapping from $\mathcal{ES}$ defined above:

Theorem 4.2 *Let δ be any program, α be any sentence of $\mathcal{ES}$, and σ be any situation term of the situation calculus.*
Then $\models_{\mathrm{FOL}} \mathit{D\tilde{o}}(\delta, \alpha)[\sigma] \;\equiv\; \exists s.\, \mathit{Do}(\delta, \sigma, s) \wedge \alpha[s]$.

This is proved by induction over δ, the only troublesome case being for the nondeterministic iteration, δ^*.

5 Mapping from the situation calculus

When mapping the situation calculus into $\mathcal{ES}$, the main issue is the treatment of quantified situations. While simple formulas can be translated using the inverse of * from Section 3, a sentence like $\forall s \exists s'.s \sqsubseteq s' \wedge (s \neq s') \wedge F(s) \equiv F(s')$, which says that from every situation another situation is reachable that agrees on the truth value of F, has no counterpart in $\mathcal{ES}$.

To deal with situation-calculus sentences like these, our proposal is to encode *action sequences* in second-order $\mathcal{ES}$. Let *nil* be a (rigid) constant and *seq* a (rigid) binary function symbol of $\mathcal{ES}$. We define $\mathit{ActionSeq}(x)$ as an abbreviation for

$$\forall Q.\, Q(\mathit{nil}) \;\wedge\; \forall y, a.(Q(a) \supset Q(\mathit{seq}(a, y))) \supset Q(x).$$

In the situation calculus, a fluent like $F(\mathit{do}(\alpha, \mathit{do}(\beta, S_0)))$ actually means that F holds after doing β then α. So the actions are in reverse order in a situation term. It is therefore useful to be able to reverse a sequence as follows:

Definition 5.1 *Rev*$(x, y) \stackrel{def}{=} \forall R(\ldots \supset R(x, nil, y))$*, where the ellipsis stands for the universal closure of*

$$R(nil, x, x) \wedge [R(x, seq(a, y), z) \supset R(seq(a, x), y, z)].$$

Next we define what it means for a formula α to be true after a sequence of actions.

Definition 5.2 *After*$(s, \alpha) \stackrel{def}{=} \forall P.(\ldots \supset P(s))$ *where the ellipsis stands for the universal closure of*

$$\Box\{(\alpha \supset P(nil)) \wedge ([a]P(x) \wedge Rev(seq(a, x), y) \supset P(y))\}$$

The fluent $\neg Broken(do(repair, do(drop, S_0)))$ will then be translated as $After(seq(repair, seq(drop, nil)), \neg Broken)$. In order to deal with quantification over situations, we define $\leq$ as a relation over action sequences, similar to the situation calculus relation $\sqsubseteq$.

Definition 5.3 $(x \leq x') \stackrel{def}{=} \forall Q.(\ldots \supset Q(x, x'))$ *where the ellipsis stands for the universal closure of*

$$Q(nil, x) \wedge (Q(x, y) \supset Q(seq(a, x), seq(a, y)))$$

The formula $\forall s \exists s'.s \sqsubset s' \wedge (s \neq s') \wedge F(s) \equiv F(s')$ of the situation calculus will then be translated as

$$\forall x.\, ActionSeq(x) \supset \exists x'.\, ActionSeq(x') \wedge (x \leq x') \wedge (x \neq x') \wedge [After(x, F) \equiv After(x', F)].$$

The final problem we must deal with concerns unrestricted quantification over initial situations. To simplify things, we restrict ourselves to $\mathcal{SC}^r$, the *rooted* situation calculus, where all situation quantification is of the form $\forall s.\ \sigma \sqsubseteq s \supset \alpha$, for some situation term σ. Note that this is no restriction at all for the non-epistemic situation calculus, since $\forall s.\alpha$ is equivalent to $\forall s.S_0 \sqsubseteq s \supset \alpha$. We handle the *Knows* operator separately.

To ease the translation, we assume that situation-calculus formulas are in the following normal form: all arguments of predicates and functions are variables; equality expressions have the form $(x = t)$, where x is a variable and t is a term mentioning at most 1 function symbol. It is easy to show that every formula of $\mathcal{SC}^r$ is equivalent to one in this normal form.

The mapping from the situation calculus into $\mathcal{ES}$ begins with a mapping $^\bullet$ from situation terms into sequences of actions in $\mathcal{ES}$. We assume that there is a countably infinite set V_s of extra object variables in $\mathcal{ES}$ for this purpose. For any situation variable s, we assume that $s^\bullet \in V_s$ and that $^\bullet$ is 1-to-1 and onto wrt V_s when restricted to situation variables. We also have that $S_0^\bullet = now^\bullet = nil$, and that $do(a, s)^\bullet = seq(a, s^\bullet)$. We extend $^\bullet$ to formulas as follows:

Definition 5.4 *Let α be any formula of $\mathcal{SC}^r$ in normal form. Then its translation into $\mathcal{ES}$, $\alpha^\bullet$, is defined inductively by:*

1. $G(x_1, \ldots, x_k)^\bullet$*, where G is a rigid predicate or rigid second-order variable, is* $G(x_1, \ldots, x_k)$*;*
2. $F(x_1, \ldots, x_k, s)^\bullet$*, where F is a fluent predicate or fluent second-order variable, is* $After(s^\bullet, F(x_1, \ldots, x_k))$*;*
3. $(x = g(x_1, \ldots, x_k))^\bullet$*, where g is a rigid function, is* $(x = g(x_1, \ldots, x_k))$*;*
4. $(x = f(x_1, \ldots, x_k, s))^\bullet$*, where f is a fluent function, is* $After(s^\bullet, x = f(x_1, \ldots, x_k))$*;*
5. $(s = \sigma)^\bullet$ *is* $(s^\bullet = \sigma^\bullet)$*;*
6. $(\sigma_1 \sqsubseteq \sigma_2)^\bullet$ *is* $(\sigma_1^\bullet \leq \sigma_2^\bullet)$*;*
7. $(\neg\alpha)^\bullet$ *is* $\neg\alpha^\bullet$*;*
8. $(\alpha \wedge \beta)^\bullet$ *is* $\alpha^\bullet \wedge \beta^\bullet$*;*
9. $(\forall x.\alpha)$ *is* $\forall x.\alpha^\bullet$ *for an object or action variable* x*;*
10. $(\forall s.\sigma \sqsubseteq s \supset \alpha)^\bullet$ *is* $\forall s^\bullet.\, ActionSeq(s^\bullet) \wedge \sigma^\bullet \leq s^\bullet \supset \alpha^\bullet)$*;*
11. $(\forall V.\alpha)^\bullet$ *is* $\forall V.\alpha^\bullet$ *for a second-order variable* V*;*
12. $Knows(\alpha, s)^\bullet$ *is* $After(s^\bullet, Know(\alpha^\bullet))$.

To show correctness of the translation, we need two more axioms Ψ, which are the universal closure of the following:
$seq(a, x) \neq nil$;
$seq(a, x) = seq(a', x') \supset a = a' \wedge x = x'$.

Theorem 5.5 *Let α be a sentence of $\mathcal{SC}^r$ in normal form. Then* $\Sigma \cup \Upsilon \models_{\mathrm{FOL}} \alpha$ *iff* $\Psi \models \alpha^\bullet$.

6 Related Work

The situation calculus, which has been the sole focus of this paper, is of course not the only language for reasoning about action. Over the years, there have been many other proposals such as the event calculus [Kowalski and Sergot 1986], Sandewall's features and fluents [Sandewall 1994], or the fluent calculus [Hölldobler and Schneeberger 1990], to name but a few. What distinguishes the situation calculus perhaps most from its alternatives is that it admits an elegant *monotonic* solution to the frame problem [Reiter 1991], which is shared to our knowledge only by its close relative, the fluent calculus [Thielscher 1999]. However, the fluent calculus also has situation terms as part of the language and hence suffers from the same problems which $\mathcal{ES}$ tries to address.

$\mathcal{ES}$ itself has much in common with dynamic logic [Pratt 1976; Harel 1984], which has also been applied to reasoning about action [Castilho, Gasquet, and Herzig 1999]. A new feature of $\mathcal{ES}$, compared to dynamic logic, are quantified modalities as in $\forall a.[a]\alpha$, which are key in expressing successor state axioms, among other things.

Dynamic logic has been combined with epistemic logic [Herzig et al. 2000; Demolombe 2003]. In the language of [Herzig et al. 2000], it is possible to express things like $[backward]\,[sonar]\,Know(Close)$ using an almost identical syntax and where *Know* also has a possible-world semantics. Despite such similarities, there are significant differences, however. In particular, their language is propositional. Consequently, there is no quantification over actions. Demolombe (2003) proposes an axiomatic translation of parts of the epistemic situation calculus into modal logic. While his modal language is first-order, he does not consider quantified modalities.

Although they do not consider epistemic notions, the work by [Blackburn et al. 2001] is relevant as it reconstructs a version of the situation calculus in Hybrid Logic [Blackburn et al. 2001], a variant of modal logic which was inspired by the work on tense logic by Prior [Prior 1967]. In a sense, though, this work goes only part of the way as an explicit reference to situations within the logic is retained. To us this presents

a disadvantage when moving to an epistemic extension. The problem is that the epistemic situation calculus requires us to consider uncountably many situations, which precludes a substitutional interpretation of quantification.

7 Conclusion

In this paper, we have substantiated the claim made informally in a previous paper that $\mathcal{ES}$ is a fragment of the situation calculus that admits an intuitive possible-world semantics. We argued that this fragment is expressive enough to represent basic action theories by presenting a simple example involving the action, knowledge, and sensing of a robot. We also showed how a version of the *Do* operator of the Golog language could be accommodated within the language. Finally, we proved that by using second-order encodings of sequences, virtually all of the situation calculus can be represented in $\mathcal{ES}$. There are a number of directions for future research, but perhaps the most immediate would be to implement a new version of Golog based on $\mathcal{ES}$ and put it to work.

7.1 Acknowledgments

We thank the anonymous reviewers for their helpful comments.

References

[Blackburn et al. 2001] P. Blackburn, J. Kamps, and M. Marx, Situation calculus as hybrid logic: First steps. In P. Brazdil and A. Jorge (Eds.) *Progress in Artificial Intelligence,* Lecture Notes in Artificial Intelligence 2258, Springer Verlag, 253-260, 2001.

[Castilho, Gasquet, and Herzig 1999] M. A. Castilho, O. Gasquet, and A. Herzig, Formalizing Action and Change in Modal Logic I: the frame problem. *Journal of Logic and Computation*, 9(5), 701–735, 1999.

[Demolombe 2003] R. Demolombe, Belief change: from Situation Calculus to Modal Logic. *IJCAI Workshop on Nonmonotonic Reasoning, Action, and Change (NRAC'03),* Acapulco, Mexico, 2003.

[Enderton 1972] H. Enderton, *A Mathematical Introduction to Logic*, Academic Press, New York, 1972.

[Harel 1984] D. Harel, Dynamic Logic, in D. Gabbay and F. Guenther (Eds.), *Handbook of Philosophical Logic*, Vol. 2, D. Reidel Publishing Company, 497–604, 1984.

[Herzig et al. 2000] A. Herzig, J. Lang, D. Longin, and T. Polacsek, A logic for planning under partial observability. In *Proc. AAAI-2000*, AAAI Press.

[Hölldobler and Schneeberger 1990] Steffen Hölldobler and Josef Schneeberger. A new deductive approach to planning. *New Generation Computing*, 8:225–244, 1990.

[Kowalski and Sergot 1986] Robert Kowalski and Marek Sergot. A logic based calculus of events. *New Generation Computing*, 4:67–95, 1986.

[Hughes and Cresswell, 1968] G. Hughes, and M. Cresswell, *An Introduction to Modal Logic*, Methuen and Co., London, 1968.

[Lakemeyer and Levesque, 2004] G. Lakemeyer and H. J. Levesque, Situations, si! Situation Terms, no!. In *Nineth Conf. on Principles of Knowledge Representation and Reasoning (KR2004)*, AAAI Press, 2004.

[Levesque *et al*, 1998] H. J. Levesque, F. Pirri, and R. Reiter, Foundations for the situation calculus, Linköping Electronic Articles in Computer and Information Science, **3**, 1998, available at http://www.ep.liu.se/ea/cis/1998/0 18/ .

[Levesque and Lakemeyer 2001] H. J. Levesque and G. Lakemeyer, *The Logic of Knowledge Bases*, MIT Press, 2001.

[Levesque et al. 1997] H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. B. Scherl., Golog: A logic programming language for dynamic domains. *Journal of Logic Programming,* **31**, 59–84, 1997.

[McCarthy and Hayes 1969] J. McCarthy and P. J. Hayes, Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer, D. Mitchie and M. Swann (Eds.) *Machine Intelligence 4,* Edinburgh University Press, 463–502, 1969.

[Pratt 1976] V. R. Pratt, Semantical considerations on Floyd-Hoare logic. In *Proc. 17th Ann. IEEE Symp. on Foundations of Computer Science,* IEEE Computer Society Press, 109–121, 1976.

[Prior 1967] A. Prior, *Past, Present and Future.* Oxford University Press, 1967.

[Reiter 1991] Ray Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation*, pages 359–380. Academic Press, 1991.

[Reiter 2001] R. Reiter, *Knowledge in Action: Logical Foundations for Describing and Implementing Dynamical Systems*. MIT Press, 2001.

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

[Scherl and Levesque, 2003] R. B. Scherl and H. J. Levesque, Knowledge, action, and the frame problem. *Artificial Intelligence* 144(1-2), 1-39, 2003.

[Thielscher 1999] Michael Thielscher. From situation calculus to fluent calculus: State update axioms as a solution to the inferential frame problem. *Artificial Intelligence*, 111(1–2):277–299, 1999.