Indirect and Conditional Sensing in the Event Calculus

Jeremy Forth
Department of Computing
Imperial College, London, SW7 2BZ, UK
jforth@imperial.ac.uk

Abstract

Controlling the sensing of an environment by an agent has been accepted as necessary for it's effective operation. Usually, however, agents operate in partially observable domains where not all parameters of interest are accessible for direct sensing. Sensing actions must then be chosen for what they will reveal indirectly, through an axiomatized model of the domain causal structure. This indirect form of sensing has received somewhat less attention in the literature. This article shows how sensing can be chosen so as to acquire and use indirectly obtained information to meet goals not otherwise possible. Event Calculus is extended with a knowledge formalism, and used to show how inferring unknown information about a domain leads to conditional sensing actions.

1 Integrating Knowledge with Event Calculus

To perform effectively in most practical domains, a rational agent must possess the ability to reason about, and create plans for sensing, conditional, and knowledge-producing actions. This requires the agent to not only reason about the state of objects in the domain, but also reason about the agent's own knowledge about the state of the domain [Moore, 1984]. Acquisition of knowledge may be achieved by direct sensing of the environment, or alternatively, indirectly through inference using existing or newly acquired knowledge.

Providing an agent with the capability to reason about its knowledge of the environment requires that we provide a theory that can represent and reason about that part of itself devoted to describing the environment. We augment domain fluents with meta-level knowledge fluents which at any given time point are semantically attached to the corresponding domain fluent using the knowledge axioms.

The language used to represent knowledge here [Turner, 1990] is a self-referential (amalgamated) language permitting a predicate to take a formula as an argument through the use of a naming relation to convert the formula to a unique term. For simplicity, we adopt the convention of sentences naming themselves. Using such a language, very general expressions are possible, such as those describing incomplete information.

Shown below are axioms employed in a first-order self-referential theory of knowledge similar to modal logic's S4 system. Event Calculus as adopted here [Shanahan, 1999a] uses the distinguished predicate HoldsAt to represent the state of all time varying fluents, including knowledge fluents, all knowledge axioms must be defined within this predicate.

$$HoldsAt[k(f_1 \rightarrow f_2) \rightarrow [k(f_1) \rightarrow k(f_2)]] \qquad \text{(K1)}$$

$$HoldsAt[\forall x k(f) \rightarrow k(\forall x\ f)] \qquad \text{(K2)}$$

$$HoldsAt[k(f) \rightarrow f] \qquad \text{(K3)}$$

$$HoldsAt[k(f) \rightarrow k(k(f))] \qquad \text{(K4)}$$

$$if\ \mathit{PC} \vdash HoldsAt(f,t)\ \ then\ \ HoldsAt(k(f),t) \qquad \text{(K5)}$$

$$HoldsAt[kw(f) \stackrel{\text{def}}{\equiv} [k(f) \vee k(\neg f)]] \qquad \text{(K6)}$$

In order to reason about knowledge, we must define inference rules that will apply to formulae within the temporal structure:

$$HoldsAt(f_1, \rightarrow f_2, t) \rightarrow [HoldsAt(f_1,t) \rightarrow HoldsAt(f_2,t)] \qquad \text{(KEC1)}$$

$$\forall x\ HoldsAt(f_1,t) \rightarrow HoldsAt(\forall x\ f_1, t) \qquad \text{(KEC2)}$$

$$HoldsAt(\neg f, t) \equiv \neg HoldsAt(f,t) \qquad \text{(NEG)}$$

2 Acting in Causal Domains

Sensing actions can be axiomatized in a domain independent way by making an assumption that all fluents can be sensed directly. The effect of a pure sensing action is limited, by definition, to changing the agent's knowledge base. There will not be any effect on the domain. Clearly, in many practical situations, sensing does impact the environment, but this will be handled as a refinement to the pure sensing action defined below. A function *sense* is introduced, mapping a fluent to an action. The sensing action is defined as follows.

$$Initiates(sense(f_p), kw(f_p), t) \tag{SA1}$$

$$Possible(sense(f), t) \leftarrow HoldsAt(observable(f), t) \tag{SP2}$$

Many actions have effects on the domain as well as on the agent's knowledge of the domain's state. An agent may come to know the status of a door by sensing, or alternatively by performing an *OpenDoor* action. Axioms KE1 and KE2 describe what the agent will come to know by performing actions.

$$Initiates(a, k(f_p), t) \leftarrow [Initiates(a, f_p, t) \wedge \neg Terminates(a, f_p, t)] \tag{KE1}$$

$$Terminates(a, k(f_p), t) \leftarrow [Terminates(a, f_p, t) \wedge \neg Initiates(a, f_p, t)] \tag{KE2}$$

Finally, we introduce the function *if* (from fluent and action pairs, to action) to represent conditional actions, executed just if the conditional fluent holds.

$$Initiates(if(f_1, a), f_{2p}, t) \leftarrow [Initiates(a, f_{2p}, t) \wedge HoldsAt(f_1, t)] \tag{CA1}$$

$$Terminates(if(f_1, a), f_{2p}, t) \leftarrow [Terminates(a, f_{2p}, t) \wedge HoldsAt(f_1, t)] \tag{CA2}$$

$$Possible(if(f, a), t) \leftarrow HoldsAt(kw(f), t) \tag{CP1}$$

2.1 Example 1

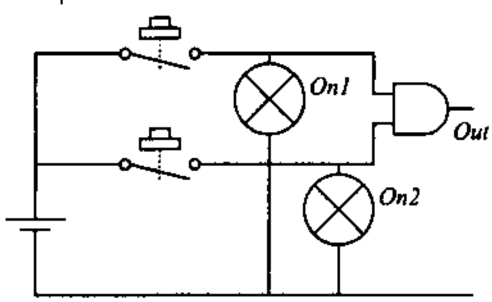

Figure 1: Unobservable output.

To illustrate the use of indirect sensing, we use the domain of Figure 1. Sensing actions are chosen to allow an agent to infer the value of *Out.* Sensing is necessary on just one of the two inputs, as long as this input is found to be inactive. If found to be active, then a second sensing action must take place. Either way, the goal of knowing whether the output is active is met.

There are two primary fluents: *On1* and On2, both of which can be directly observed. The output is a derived fluent, which, for the sake of example is specified to be unobservable. The goal is a knowledge goal stating that the robot knows if the output is active or not: $HoldsAt(kw(Out), 3)$ To describe the environment's ramification, we use a state constraint, and for the sake of simplicity also provide an associated knowledge constraint:

$$HoldsAt(Out, t) \equiv [HoldsAt(On1, t) \wedge HoldsAt(On2, t)] \tag{SC2}$$

$$HoldsAt(k(Out), t) \equiv [HoldsAt(k(On1), t) \wedge HoldsAt(k(On2), t)] \tag{KSC2}$$

It is necessary to state that the fluents *On1* and *On2* are observable:

$$HoldsAt(observable(On1), t) \tag{FD1}$$

$$HoldsAt(observable(On2), t) \tag{FD2}$$

A two-action narrative is defined; a sensing action of fluent *On1* followed by a conditional sensing action of fluent On2, contingent upon *On1* having been found active:

$$Happens(sense(On1), 1) \tag{N3}$$

$$Happens(if(On1, sense(On2)), 2) \tag{N4}$$

Using appropriate Uniqueness-of-names axioms, and a circumscriptive solution to the frame problem,

CIRC[SA1,CA1-2; *Initiates, Terminates*], CIRC[N3-4; *Happens*], and CIRC[CP2,SP1,TP1; *Possible*]:

we can prove the goal *HoldsAt(kw(Out),3).* This result demonstrates that the conditional sensing plan for knowledge acquisition in the domain of Figure 1 will always yield information about the state of the unobservable output. However, depending on the state of the first sensed fluent *On1,* the agent may not have to sense *On2* at all.

3 Conclusion

We have introduced a knowledge formalism into Event Calculus to allow an agent to represent it's knowledge of an environment sufficiently to reason about sensing, conditional actions, and inference of environmental unknowns through state constraints. It is hoped that this will form the basis for a comprehensive account of knowledge producing actions in further work.

References

[Moore, 1984] Robert C. Moore. A formal theory of knowledge and action. In *Formal Theories of the Commonsense World.* Ablex Publishing Corp., Norwood, New Jersey, 1984.

[Shanahan, 1999a] Murray Shanahan. The event calculus explained. *Lecture Notes in Computer Science,* 1600:409-434, 1999.

[Turner, 1990] R.Turner. *Truth and Modality for Knowledge Representation.* Pitman Publishing: London, 1990.