Knowledge as a Tool in Motion Planning under Uncertainty

Ronen I. Brafman
Robotics Laboratory
Stanford University
Stanford, CA 94305
brafman@cs.stanford.edu

Jean-Claude Latombe
Robotics Laboratory
Stanford University
Stanford, CA 94305
latombe@cs.stanford.edu

Yoram Moses*
Dept. of Applied Math
Weizmann Institute of Science
Rehovot, 76100 Israel
yoram@wisdom.weizmann.ac.il

Yoav Shoham
Robotics Laboratory
Stanford University
Stanford, CA 94305
shoham@cs.stanford.edu

Abstract

Inspired by the success of the distributed computing community in applying logics of knowledge and time to reasoning about distributed protocols, we aim for a similarly powerful and high-level abstraction when reasoning about control problems involving uncertainty. Here we concentrate on robot motion planning, with uncertainty in both control and sensing. This problem has already been well studied within the robotics community. Our contributions include the following:

- We define, a new, natural problem in this domain: obtaining a sound and complete termination condition, given initial and goal locations.
- We define a high-level language, a logic of time and knowledge, to reason about motion plans in the presence of uncertainty, and use it to provide general conditions for the existence of sound and complete termination conditions for a broad class of motion plans.
- We characterize the optimal sound termination conditions for the general problem, relate them to a class of fundamental knowledge based protocols and provide a natural example of knowledge based protocols lacking a canonical implementation.[1]

*Currently on sabbatical at the Oxford University Computing Laboratory, Oxford OX1 3QD England

[1]The first part of this paper generalizes results of a previous paper by Brafman, Latombe and Shoham ([BLS93]). Sections 4.3 and 5 contain new material.

1 Introduction

Much research carried on in computer science in general, and AI in particular, concerns the development of powerful abstractions, and their application to problems of interest. In the context of this article, of particular note is the application of logics of knowledge in distributed computing (e.g., [HM90]). The essential insight behind that line of research was that a formal notion of "knowing," developed initially in philosophy [Hin62] and later imported to AI [Moo85], can be coherently and usefully applied to reasoning about (and later also designing) distributed protocols. The reasons for the success of this approach include:

Intuitiveness: The high-level language supported statements of the sort "processor A doesn't know that processor B is faulty," which are precisely the type employed informally by people reasoning about the domain.

Groundedness: The formal notion of knowledge was anything but vague; it was defined precisely in terms of the underlying protocol.

Abstraction and Generality: In principle, the notion of knowledge was dispensable. However, the analysis in terms of knowledge homed in on the essential notion, the knowledge available to the various processors at different points in time, and allowed one to abstract away from the details of how the particular physical protocol implemented that knowledge. This knowledge-level abstraction made it possible to analyze (and later also design) protocols even before their physical implementation was specified; in fact, the same knowledge-level protocol could be implemented differently, without affecting the high-level analysis.

While logics of knowledge have been widely used in AI (e.g., to model human-computer interaction, distributed planning, and nonmonotonic logics), they have so far not been applied in a similar fashion, as a knowledge level corresponding to some specific concrete system. Two exceptions do come to mind – Levesque's knowledge-level analysis of databases [Lev84], and Rosenschein and Kaelbling's Situated Automata [Ros85, KR90]. We claim, however, that there is a much wider arena in which the lessons from distributed computing can be applied, namely planning and control in the presence of uncertainty. In this article we take one step towards exploring this arena, concentrating on robot motion planning.

Robot motion planning with uncertainty is a well researched area [LPMT84, Can89, Erd86, Lat91, LLS91]. The uncertainty in that domain can arise from several sources, including partial information about the location of various objects, sloppy control, and noisy sensing. We argue that this domain exhibits all the 'right' properties: (1) One naturally analyzes the situation by saying that "the robot knows that it is at the goal, since it knows that the current reading could only have been obtained if it were either at the goal or beyond the wall, and it knows its motion plan could not possibly have taken it behind the wall"; (2) the notion of knowledge can be grounded precisely in the motion plan of the robot, as well as some additional parameters such as the slop in control and the noise in sensing.

It would have been convenient to start with a given class of motion planning problems, and delve directly into their knowledge-level analysis. However, we were surprised to find that, although much related research has been conducted in robotics, the simple question

we would like to pursue has not been addressed. A typical question asked in robotics is "Given that the robot must end up in a particular region, and given bounds on the slop in control and noise in sensing, what is the biggest initial area from which the robot can start, and still be guaranteed to arrive at the goal and recognize that it is at the goal?" This initial area is called the *pre-image* of the goal. In a multi-step motion plan, this question is repeated in a backward-chaining fashion, leading to the method of *pre-image backchaining* [LPMT84]. In contrast, we consider fixed initial and goal regions and a class of simple motion commands 'Go in direction D, until the termination condition, T, is satisfied'. D can be seen as responsible for reaching the goal, while T is responsible for recognizing it. The seemingly more basic question we ask is: "Given a fixed D, and given bounds on the slop in control and noise in sensing of the robot, does there exist a *good* definition of T?" Of course, one could interpret *good* in many ways. We will interpret it by appealing to standard computer-scientific notions; we will be interested in termination conditions that are *sound* and *complete*, that is, ones that guarantee that if the robot stops, it only stops at the goal, and that it does eventually stop. And while researchers in motion planning (e.g. [LLS91]) gave examples indicating that careful design of a termination condition can much enhance the robot's ability to recognize the goal, we are the first to offer general results on this question.

Before introducing a high-level language, we will give more feel for the problem by examining a simple one-dimensional domain. This simple path-planning problem in $\mathbb{R}^1$, taken from [Lat91], will be used later, to illustrate the various definitions and results.[2]

Example 1 *Assume that our robot is a point moving forward along the positive reals, starting at 0; it moves continuously at finite velocity, until the termination condition is satisfied, at which point it stops. The goal is the interval* $[2,4]$. *There is a position sensor with a sensing uncertainty of 1, so that if the robot is at location* ℓ, *its sensor may indicate any value between* $\ell - 1$ *and* $\ell + 1$.

In the following let r *denote the current position reading of the robot. Clearly* '$r > 1$' *is a complete termination condition, but not a sound one. Similarly,* '$r = 3$' *is a sound termination condition, but not a complete one (readings need not be continuous: we may have a sequence of readings that are accurate until we reach* 2.5, *at which point they might become consistently off by +1, i.e., start from* 3.5 *and grow). Somewhat surprisingly, there exists a termination condition that is both sound and complete, e.g.,* '$r \in [3,5]$'. *To see that this is the case, notice that this termination condition must evaluate to* true *by the time we reach the position 4.*

In the following sections we will consider general motion planning, define a (fairly standard) logic of knowledge and time for reasoning about it, and then, for a broad class of motion planning problems in $\mathbb{R}^n$, provide a knowledge-level characterization of the conditions in which sound and complete termination conditions exist. To answer the general problem we will define a fixpoint operator that enables the characterization of sound termination conditions that are optimal in some sense. We end by showing that the problem of optimal

[2] Additional examples appear in [BLS93].

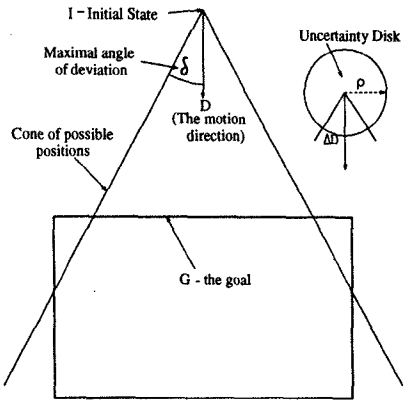

Figure 1: An example domain

termination conditions provides the first natural example of knowledge-based protocols that have no single canonical implementation ([HF89]) and that this problem can be phrased as a problem of generating the optimal implementation of a knowledge based protocol.[3]

2 Termination Conditions in Motion Planning

We introduce a motion planning domain, with particular types of sensing and control uncertainty, and formally define the problem of the existence of sound and complete termination conditions. Our model is based on [LPMT84].

Figure 1 will help us illustrate the problem we are investigating. Starting from any position within a given set $I \subset \mathbb{R}^n$ (a singleton in Figure 1), the robot is commanded to move in a (predetermined) direction D. The goal is to arrive at the region $G \subset \mathbb{R}^n$, the *goal region*, area, and stop there. The discussion in this paper will be limited to compact goal regions.

Our control of the robot's motion is imperfect: at each point the tangent to the robot's path may deviate from D by up to δ, the *control uncertainty*. While moving, the robot senses its location via an imperfect position sensor, which returns a position reading that may be different by a distance of up to ρ from the *actual* position. Based on this reading the robot decides whether to stop or continue its motion. But while the robot's motion is continuous, the position readings need *not* be continuous.[4]

The bounded uncertainty in control constrains the robot to remain within a set of cones of possible positions defined by the initial states, by D and by δ (as illustrated in Figure 1). At each position, q, the robot's sensors supply a position reading, r, which must be within a disk of radius ρ, centered at q. Consequently, given a position reading, r, the actual position, q, is also within a disk of radius ρ, centered in r, defining the *sensing uncertainty*. Any subset

[3]Only that section assumes familiarity with the notions of knowledge-based protocols and their implementation.

[4]One reason for choosing to have this flexibility is that it enables modelling robots that do not continuously evaluate their readings.

of this disk outside the cone of possible positions can be eliminated as candidates for the actual position.

We use the term **motion planning instance**, denoted by ξ, to refer to the domain specified by δ, ρ, I, and G. The **termination condition**, T, is a (total) boolean function on the set of possible readings. The first time it evaluates to *true* the robot stops. In our discussion we assume no knowledge regarding the robot's velocity, except that it must exceed some $\epsilon > 0$ as long as the robot has not stopped.

Returning to Example 1, we see that the motion is in $\mathbb{R}$, the motion direction is $+$, the goal is $[2,4]$, there is no control uncertainty (although, as we mentioned, the velocity is unknown), but the sensing uncertainty is $\rho = 1$.

We shall use annotated trajectories to describe the movement of a robot. Formally, an *annotated trajectory* is a pair $\tau = (Q, V)$, where Q is a differentiable function from $[0, \infty)$ to $\mathbb{R}^n$, and V is a (not necessarily continuous) function from $[0, \infty)$ to $\mathbb{R}^n$. The function Q describes the robot's position at any instant of time, while V describes the sensor readings. A *motion command* is a pair $M = (D, T)$, where D is a direction and T is a termination condition.

Definition 1 *We say that the annotated trajectory $\tau = (Q, V)$ is* consistent *with a given motion planning instance $\xi = (\delta, \rho, I, G)$ and a motion command $M = (D, T)$, if the following properties are satisfied:*

- *$Q(0) \in I$;*
- *$\forall t \in [0, \infty) : \ |V(t) - Q(t)| \leq \rho$;*
- *$\forall t \in [0, \infty) : \ |dQ(t) - D| \leq \delta$, where $dQ(t)$ is the direction of the tangent to Q at t;*
- *$\forall t \in [0, \infty) : \ $ if $T(V(t))$ is true, then $\forall t' > t \ Q(t') = Q(t)$.*
- *$\exists \epsilon > 0, \forall t < \hat{t}$, where $\hat{t} = inf\{t \mid T(V(t))\}$, we have $||\dot{Q}(t)|| > \epsilon$.*

Intuitively, Definition 1 states that the trajectory starts from a "legal" initial state, the accuracy of the sensor conforms to the specifications, the direction of movement is within the allowed difference δ from the direction specified by the motion command, the termination condition is obeyed, and finally that before it halts, the robot's speed always exceeds a certain nonzero lower-bound of ϵ.

Definition 2 *Given a motion planning instance $\xi = (\delta, \rho, I, G)$ and a motion direction D, a termination condition T is* **sound** *if for every annotated trajectory $\tau = (Q, V)$ consistent with $M = (D, T)$ and ξ we have: $\forall t \, [T(V(t)) \rightarrow (Q(t) \in G)]$. The condition T is* **complete** *if for every annotated trajectory consistent with (D, T) and ξ it is the case that $T(V(t))$ holds for some $t \geq 0$.*

Thus, a termination condition is sound if it prescribes stopping only when the robot is in the goal, and it is complete if the robot is always guaranteed to eventually stop. The following is a precise formulation of the problem we wish to investigate:

Given a motion planning instance and a motion direction D, does a sound and complete termination condition exist?

In [BLS93] a geometric property of the domain was defined that is sufficient, and under certain conditions necessary, for the existence of a sound and complete termination condition. Our proof employed purely geometric reasoning, and is constructive, showing how to build a sound and complete termination condition, when one exists. In the next section we define a basic framework that will allow us to perform a general study of the design of optimal termination conditions in motion planning by reasoning about the robot's knowledge.

3 Knowledge-Level Formalization

While new, the geometric results of [BLS93] were restricted to a limited class of sensors (e.g., position sensors exhibiting constant disk-like uncertainty), constant control uncertainty and limited sets of initial states. In order to overcome these limitations, a framework for reasoning about uncertainty in motion planning using a formal notion of knowledge was introduced in that paper. This framework, motivated by work in distributed systems ([HM90, FHMV94]), is described next.

3.1 Runs and systems

We can view our endeavor as the investigation of a class of two-player protocols (see [TMG88]). Our players are the robot, which follows the motion command, and the environment, which decides nondeterministically how the robot actually moves and what it senses, within given margins. We shall thus consider an instantaneous description of the state of affairs to consist of a state for the environment and a state for the robot. Let us make this more precise.

Definition 3 *Let $\mathcal{R}$ denote the set of local states of the robot, while $\mathcal{E}$ denotes the set of possible (local) states of the environment. A* **global state** *is a pair (e,r), where $e \in \mathcal{E}$ and $r \in \mathcal{R}$. We denote the set of global states by $\mathcal{G}$.*

In the domain of the previous section the local state of the robot consisted of its current sensor reading, while the local state of the environment consisted of the robot's *actual* position.

The behavior of the robot and the environment over time will be captured by considering *runs*, which will associate a global state to every time instant.

Definition 4 *A* **run** R *is a function from $[0,\infty)$ to $\mathcal{G}$. We identify a* **system** *with a set of runs.*

We remark that this definition of runs differs from the standard definitions [FHMV94] in that we assume that time is continuous rather than discrete. A system corresponds to some subset of the set of possible runs, those runs that describe the behaviors we would like to model. For example, a natural system that we would associate with a given motion planning

instance ξ and a motion command $M = (D, T)$, is the set of all runs consistent with ξ and M. We denote this system by $S(\xi, M)$. The system $S(\xi, M)$ captures the robot's set of possible behaviors when the environment's behavior conforms with ξ and the robot's speed is always greater than some $\epsilon > 0$ as long as the termination condition does not hold.

3.2 A language and its semantics

We now present a language for reasoning about the design of termination conditions in motion planning. This language contains temporal and epistemic modal operators and has intuitive semantics.

We assume some propositional language $\mathcal{L}$. In particular, we will assume the existence of a primitive proposition g and that for every termination condition T of interest, $\mathcal{L}$ contains a primitive proposition T. Given the set $\mathcal{G}$ of global states of a system S, an *interpretation* π for $\mathcal{L}$ in $\mathcal{G}$ is a function assigning a truth value to every primitive proposition of $\mathcal{L}$ at every state in $\mathcal{G}$. We call a pair $\mathcal{I} = (S, \pi)$, consisting of a system S and an interpretation π over the global states in S, an *interpreted system.* We can define the satisfaction of formulas with respect to interpreted systems. Specifically, we say that a formula α is satisfied at a global state s in interpreted system $\mathcal{I}$ when:

- $\mathcal{I}, s \models p$ for a primitive proposition p if $\pi(s, p) = true$;
- $\mathcal{I}, s \models \neg\alpha$ if and only if $\mathcal{I}, s \not\models \alpha$; and
- $\mathcal{I}, s \models \alpha \wedge \beta$ if and only if $\mathcal{I}, s \models \alpha$ and $\mathcal{I}, s \models \beta$.

In general, we assume that the interpretation function π is 'natural', e.g., the proposition g, denoting a goal state, will be satisfied *exactly* by those states in which the position is part of the goal, and the proposition T will be satisfied exactly when the condition T holds. Given that we assume the interpretation π to be fixed, we shall refrain from describing π explicitly in every system we consider. We shall talk about a run R of a system S as also being a run of the interpreted system $\mathcal{I} = (S, \pi)$. We will use $\mathcal{I} \models \alpha$ when α is satisfied by all global states that occur in (runs of) $\mathcal{I}$.

Motion is closely connected with time, thus we add temporal operators. These are operators that allow us to reason about what has happened in the past or will happen in the future of a given state. We shall use the so-called *branching-time* temporal operators, that allow us to quantify over the possible futures or pasts of a given global state, and specify whether a formula of interest is satisfied at some point in the past or future, or at all points. Temporal logics of this type have been extensively investigated by various researchers. For details see, e.g., [EH85]. Rather than presenting a complete exposition of branching-time temporal operators, we shall concentrate on the two particular operators that will be used in our analysis. Other, similar, operators can be defined in an analogous fashion.

- **There exists a future, always α:** We define $\mathcal{I}, s \models \exists\Box\alpha$ to hold if there exists a run R of $\mathcal{I}$ and a time t such that $\mathsf{R}(t) = s$ and for all $t' \geq t$ we have $\mathcal{I}, \mathsf{R}(t') \models \alpha$. I.e.,

$\exists_{\Box}\alpha$ holds at a state s in system $\mathcal{I}$ if for some occurrence of s in a run R of the system, α holds at s and at all later states in the run.

- **For all pasts, previously α:** We define $\mathcal{I}, s \models \forall_{\Diamond}\alpha$ to hold if for all runs R of $\mathcal{I}$ and times t, if $\mathsf{R}(t) = s$, the there exists a time $t' \leq t$ for which $\mathcal{I}, \mathsf{R}(t') \models \alpha$. In words, $\forall_{\Diamond}\alpha$ holds at s if no matter how the robot has arrived at s, it must have previously (or presently) satisfied α .

To deal with uncertainty we define the notion of knowledge.

Definition 5 *We say that two global states s and s' are* **indistinguishable** *to the robot, denoted by $s \sim_r s'$, precisely if the local state of the robot is identical in s and s'.*

Note that $\sim_r$ is an equivalence relation.

Definition 6 *Let $\alpha \in \mathcal{L}$. The robot* **knows** *α at a state s in $\mathcal{I}$, written $\mathcal{I}, s \models K_r\alpha$ if $\mathcal{I}, s' \models \alpha$ holds for every state s' in $\mathcal{I}$ such that $s \sim_r s'$.*

From now on we shall assume that our language, $\mathcal{L}$, is closed under the temporal and epistemic operators. All definitions remain unchanged. We remark that our definition of knowledge is the standard one used for knowledge in distributed systems (see [FHMV94]), and is well-known to satisfy the modal system S5. Note that once we fix a system S of interest, the satisfaction in state s of a formula of the form $K_r\alpha$ depends only on the local state of the robot, and can be interpreted as a predicate on its local state. $K_r\alpha$ can thus be used to define a termination condition (although not necessarily an easily computable one).

4 Optimal termination conditions

We now wish to apply the framework defined in Section 3 to the analysis of termination conditions. The framework of Section 3 allows us to study the issue of termination at a fairly abstract level by investigating properties of general systems, which can correspond to quite different assumptions on our robot and his domain. Therefore, the results we shall obtain will be applicable to many concrete contexts of interest.

4.1 The motion planning context

Roughly speaking, we would like to cast the question of designing optimal termination conditions in terms of the following question:

> *Having fixed all relevant parameters other than the termination condition, when does a sound and complete termination condition exist? If a "best" termination condition exists, what form will it have?*

Because our ability to speak about arbitrary sets of runs gives great flexibility in modelling different contexts, this generalizes the question we asked in Section 2. In general, to incorporate various assumptions on the setting investigated, we consider five main parameters that can be varied in the specification of the system.

- The specified direction: While in a motion command $M = (D, T)$ we assumed a fixed direction D, we may in general imagine a more elaborate specification of direction which could depend, for example, on the robot's sensor readings and the position of the goal, or on the actual position (due, for example, to changes in the local orientation of the surface the robot is traversing).
- Knowledge about the initial position: This will be given in terms of a set of possible initial states or initial positions. In a motion planning instance ξ this is captured by I.
- Control uncertainty: In a motion instance ξ this was captured by δ, the maximal possible deviation of the actual direction from the commanded motion direction, D More generally, the control uncertainty may change, possibly growing over time, or varying depending on the robot's current position.
- Sensing uncertainty: In a motion planning instance ξ, at any position, the robot's possible position readings could be within a disk of radius ρ centered at this actual position. In general, the possible readings may depend on time and on the robot's position. For example, the robot's position readings may be less accurate the farther he is from his initial state. Sensing is also not necessarily restricted to position sensing. It could also use force sensing or employ sensing memory.
- Progress guarantees: It is usually hard to ensure nontrivial properties for a robot that is not guaranteed to move. It is therefore customary to make certain assumptions regarding the robot's progress. For example, our definition of a consistent trajectory in Section 2 required that the robot's velocity is greater than some $\epsilon > 0$ as long as it has not explicitly halted. We call this a *progress guarantee*, or a *liveness condition.* A weaker progress guarantee that is sufficient to assume in many applications of interest, is that the robot is guaranteed to move an infinite distance over an infinite amount of time, unless it explicitly halts after a finite amount of time. We shall call this condition the *weak progress guarantee.*
- The goal region G.

We call a collection of assumptions describing the above parameters a *motion planning context.* Notice that varying any of these parameters (except perhaps the goal region) will change the set of possible trajectories. Varying the goal region will change the meaning of the soundness of a termination condition. The contexts discussed in Section 2 were ones determined by a motion planning instance $\xi = (\delta, \rho, I, G)$, a fixed direction D, and the progress guarantee that in every run the robot speed was greater than some $\epsilon > 0$ as long as it has not performed an explicit halting action.

Roughly speaking, every motion planning context κ can be viewed as defining a system $\mathcal{I}(\kappa)$ consisting of all runs consistent with the assumptions defined by κ. We view this as the

system corresponding to what would happen if the robot never actually performs an explicit halting action, or perhaps follows the trivial termination condition *false*, which is never satisfied. This system can be used to study and compare the effect of applying different termination conditions in the given setting. The only assumption that will be inherent in this choice is that no aspect of the robot's movement in a given run, before the robot explicitly halts, is affected by the termination condition the robot is using.

To obtain our results at their fullest generality, we shall not commit ourselves to specific properties of the context in which the motion planning is performed. Consequently, our results will be applicable to a wide range of different contexts, as discussed above. Our strategy will be to start out from a system $\mathcal{I}$ in which the robot is not following a termination condition, and study what happens if the robot were to follow particular termination conditions with respect to that system. We call such a system $\mathcal{I}$ an *uninhibited* system, the idea being that a termination condition serves to inhibit the robot's movement. For example, in the context of motion planning instances as defined in Section 2 the uninhibited systems are systems of the form $\mathcal{I}(\xi, M) = (S(\xi, M), \pi)$, where the termination condition specified in M is *false*. We remark, however, that this need not imply that the robot is not allowed to stop moving if this is consistent with the particular progress guarantee assumption we make.

It is sometimes possible to judge one termination condition to be 'stronger' than another. Intuitively, this means that one makes us stop at least as soon as the other. We make this precise by the following definition.

Definition 7 *Given an uninhibited system $\mathcal{I}$, a termination condition T is* **as strong** *as T' with respect to $\mathcal{I}$ if*

$$\mathcal{I} \models T' \rightarrow \forall_{\diamond} T$$

Given a class $\mathcal{T}$ of termination conditions, T is **optimal** *(with respect to $\mathcal{I}$) if it is as strong as any other $T' \in \mathcal{T}$.*

One useful observation given these definitions is the following:

Lemma 1 *(a) The class of sound termination conditions with respect to a given uninhibited system $\mathcal{I}$ and goal region G has an optimum.*

(b) If a sound and complete termination condition for a system $\mathcal{I}$ and goal region G exists, then the optimal sound termination condition for $\mathcal{I}$ and G is also optimal with respect to the class of sound and complete termination conditions.

Lemma 1(b) implies that by designing an optimal sound termination conditions, we also obtain optimal a sound and complete termination condition whenever one exists.

4.2 Optimal sound and complete termination conditions

The following theorem offers an optimal representation of sound and complete termination conditions for a large class of domains.

Theorem 1 *Let $\mathcal{I}$ be an uninhibited system, let G be the goal region, and let g be a proposition that is satisfied precisely when the robot is in a goal state. Moreover, assume that $\mathcal{I} \models (\neg g \wedge \forall_{\Diamond} g) \rightarrow \exists_{\Box} \neg g$, and that a sound and complete termination condition for $\mathcal{I}$ and G exists. Then $K_r \forall_{\Diamond} g$ defines the optimal sound and complete termination condition for $\mathcal{I}$ and G.*

In words, Theorem 1 characterizes the optimal sound and complete termination condition as the condition described by $K_r \forall_{\Diamond} g$, under particular assumptions. The theorem states that this will happen in a system in which a sound and complete termination condition *exists*, provided that the condition $(\neg g \wedge \forall_{\Diamond} g) \rightarrow \exists_{\Box} \neg g$ is valid. While technically a bit stronger, this condition is implied in every context in which, whenever the robot passes through the goal region and exits from it, the robot might never enter the goal region again. As we shall see below, there is a fairly wide class of settings in which this property holds.

Discussion: There are a number of important things to note.

- **Constructive canonical form:** The theorem gives a constructive definition of a sound and complete termination condition, if one exists, so we need only check this condition to verify the existence of a sound and complete termination condition. Note that, in general, this may be computationally expensive.
- **Optimality:** For any run, this termination condition evaluates to *true* no later than any other sound and complete termination condition, i.e. it is optimal. (We note that the use of reasoning about knowledge has given rise to optimal solutions of an analogous flavor, in a very different problem domain dealing with fault-tolerant distributed systems [DM90, MT88].)
- **Generality:** The use of knowledge to characterize the termination condition allowed us to prove a fairly general result regarding an extremely wide class of motion planning contexts. As described at the beginning of Section 4, the statement of Theorem 1 does not make very specific assumptions about the control and sensing uncertainties, the shape of the set of initial positions or the goal region, or about the progress guarantees. In fact, the theorem applies even in contexts in which the various parameters are correlated in various ways. The formalism allows us to make minimal assumptions and obtain a generally applicable result.

The following lemma shows that Theorem 1 covers a fairly large natural family of contexts. It considers motion planning contexts that generalize the motion planning instances of Section 2 by relaxing the sensing uncertainty and progress guarantees. The sensing activity is assumed to be given by an arbitrary function $\Delta Q : \mathbb{R}^n \rightarrow 2^{\mathbb{R}^n}$ that, for every possible position q of the robot describes a set of possible readings. We now have:

Lemma 2 *Let κ be a motion planning context defined by parameters $\delta, \Delta Q, I, G, D$ and the weak progress guarantee, where δ, I, G, D are as in a motion planning instance, and ΔQ describes the robot's sensing uncertainty as a function of its position as described above. If the goal region G is a convex subspace of $\mathbb{R}^n$, then*

$$\mathcal{I}(\kappa) \models (\neg g \wedge \forall_{\Diamond} g) \rightarrow \exists_{\Box} \neg g.$$

Example 1 (continued) *Recall that our robot moved along the positive reals, its goal was to be in* $[2,4]$ *and its reading uncertainty was* $\rho = 1$. *The condition* $\forall_{\Diamond} g$ *is satisfied by positions* $q \in [2,\infty)$, *thus* $K_r \forall_{\Diamond} g$ *is satisfied by readings* $r \in [3,\infty)$, *corresponding to a sound and complete termination condition.*

A natural question to ask is what one gains by designing a robot with complete memory of past readings. While there are cases where this additional power may enable construction of a sound and complete termination condition when one based only on the current reading does not exist, using Theorem 1 we have been able to prove that this is not the case for the domains discussed in Section 2.

Theorem 2 *Let* κ *be a motion planning context defined by a motion planning instance* $\xi = (\delta, \rho, I, G)$ *where* G *is convex, a direction* D *and an arbitrary progress guarantee. Assume that (i) the condition* $\mathcal{I}(\kappa) \models (\neg g \wedge \forall_{\Diamond} g) \rightarrow \exists_{\Box} \neg g$ *of Theorem 1 holds for a robot with sensing history; and (ii) an optimal sound and complete termination condition* T *based on a complete reading history exists. Then there is a termination condition* T' *dependent only on the robot's* current *position reading, such that* $\mathcal{I}(\kappa) \models T \equiv T'$.

We remark that in our exposition so far we have ignored one relevant parameter, which is the structure of the domain in which the robot moves. Theorem 2 holds *only* when this domain is free of obstacles, and its proof relies on Theorem 1 together with the geometric properties of the domain.

4.3 Optimal sound termination conditions

The termination condition given in Theorem 1, while providing the optimal sound and complete termination condition *when one exists*, may fail to be even sound otherwise. One is therefore inclined to ask a slightly different question, namely, what is the optimal sound termination condition? Once an answer to this question is obtained we can answer our original problem by checking whether the optimal sound termination condition is complete, enabling a generalization of Theorem 1. As shown in Lemma 1, if a sound and complete termination condition exist, then the optimal sound termination condition must be complete. Moreover, since the condition *false* is sound, the class of sound termination conditions is never empty, so that an optimal sound condition is always guaranteed to exist. Unlike in the previous section, we now give a general characterization of the optimal sound termination condition; yet, due to the use of fixpoints in its definitions, this characterization does not have the same constructive flavor. We will have more to say on this issue later.

We close our language under a new modal operator, T.

Definition 8 *Given a system* $\mathcal{I}$, *for each* $\alpha \in \mathcal{L}$ *the formula* $\mathsf{T}\alpha$ *is satisfied by the largest*

set of states of $\mathcal{I}$ for which the following holds:[5]

$$\mathcal{I} \models \mathsf{T}\alpha \equiv K_r(\alpha \vee \forall_\Diamond(\mathsf{T}\alpha \wedge \alpha))$$

For T to be well defined we have to guarantee that there is a greatest set (i.e., greatest fixpoint of this equation). However since the mapping: $x \rightarrow K_r(\alpha \vee \forall_\Diamond(x \wedge \alpha))$ is monotone, this is the case. This definition captures the intuition that when the robot 'considers' what positions are possible given its current readings, it should discount positions that cannot be reached without previously passing through states that satisfied the termination condition.

Theorem 3 *Let κ be a motion planning context and G be a goal region, with g the proposition corresponding to the robot's being in G. Then the condition $\mathsf{T}g$ (defined with respect to $\mathcal{I}(\kappa)$) is an optimal sound termination condition for $\mathcal{I}(\kappa)$ and G.*

While the fixpoint definition may be useful for reasoning about optimal termination conditions, it does not indicate how to go about computing them. The following is a step in that direction.

Theorem 4 *Let $\mathcal{I}$ be a system. Define $\mathsf{T}_\sigma\alpha$ as follows:*

- $\mathsf{T}_0\alpha \equiv$ *true;*
- $\mathsf{T}_{\sigma+1}\alpha \equiv K_r(\alpha \vee \forall_\Diamond(\mathsf{T}_\sigma\alpha \wedge \alpha))$ *for successor ordinals;*
- $\mathsf{T}_\sigma\alpha \equiv \bigwedge_{\tau<\sigma} \mathsf{T}_\tau\alpha$ *for limit ordinals.*

There exists an ordinal σ for which $\mathcal{I} \models \mathsf{T}\alpha \equiv \mathsf{T}_\sigma\alpha$.

This theorem gives an iterative method for calculating T, albeit one that may never terminate. However, we believe that for many reasonable domains, e.g., polygonal goals and obstacles, only a small number of iterations will be needed. Support to this is lent by noticing that $\mathsf{T}_1 g \equiv K_r(g \vee \forall_\Diamond g) \equiv K_r \forall_\Diamond g$, and the latter is the termination condition used in Theorem 1. Thus, very roughly, Theorem 1 can now be understood as a claim regarding the rate of convergence of T_σ to T, a rather interesting perspective.

5 Optimal termination and knowledge-based programs

We now consider a slightly different view of the problem of designing optimal sound and complete termination conditions. This view has to do with the notion of a *knowledge-based program* and its implementations, which is described in [FHMV94] and is related to the knowledge-based protocols of [HF89]. Due to considerable technical background that would be necessary for a complete exposition, our discussion below will contain only partial details. For further technical background, the interested reader should consult [FHMV94].

[5] A more precise definition along the lines of the [HM90, FHMV94] fixpoint definition of common knowledge can be given. Adding a fixpoint operator ν to the language we would define: $\mathcal{I}, s \models \mathsf{T}\alpha$ if and only if $\mathcal{I}, s \models \nu x[K_r\alpha \vee \forall_\Diamond(x \wedge \alpha)]$.

Theorem 1 describes an optimal sound and complete termination condition in terms of the robot's knowledge about his current and past positions. In general, by depending only on the robot's local state, a termination condition can be thought of as a condition on the agent's knowledge. Can we characterize the knowledge a robot should have when halting according to an optimal termination condition? In considering this question, it is important to observe that the runs describing a robot following a termination condition $T \neq \mathit{false}$ can be quite different from those that describe the robot when its movement is not inhibited by a termination condition. Starting from an uninhibited system $\mathcal{I}$, a termination condition T induces a *different* system $\mathcal{I}[T]$ describing the runs of the robot in the same context, when it follows T. In order to study the robot's knowledge when it actually halts, we need to consider the system $\mathcal{I}[T]$. We now describe this system more formally.

For ease of exposition, we shall restrict attention in this section to systems in which for every run R and local state r, if $\hat{t} = inf\{t : \mathsf{R}(t) = r\}$ then $\mathsf{R}(\hat{t}) = r$. This condition is sensible, since it is satisfied in contexts in which the robot is able to perform only a finite number of sensor readings in a finite amount of time. Given a run R of an uninhibited system $\mathcal{I}$, and a termination condition T, let us denote by $\hat{t}(\mathsf{R})$ the time $\min_t\{T$ is true of the robot's local state at $\mathsf{R}(t)\}$. If, for no time t is T true of the robot's local state at $\mathsf{R}(t)$, then $\hat{t} = \infty$. The run $\mathsf{R}[T]$ is defined to coincide with R up to time $\hat{t}(\mathsf{R})$, at which point the robot ceases to move, and it never moves again. Given an uninhibited system $\mathcal{I}$, we say that a termination condition T *induces* the system $\mathcal{I}[T]$, where $\mathcal{I}[T]$ is the interpreted system whose set of runs is $\{\mathsf{R}[T] \mid \mathsf{R} \in \mathcal{I}\}$. With this definition, we have

Lemma 3 *Let $\mathcal{I}$ be an uninhibited system, let G be a goal region, and let T be a termination condition for $\mathcal{I}$.*

(a) T is sound for $\mathcal{I}$ and G if and only if $\mathcal{I}[T] \models T \rightarrow K_r g$; and

(b) if T is an optimal sound and complete termination condition for $\mathcal{I}$ and G, then $\mathcal{I}[T] \models T \equiv K_r g$.

Lemma 3(b) resembles Theorem 1 in that it characterizes an optimal sound and complete termination condition T in terms of knowledge. In this case, however, the characterization is in terms of the knowledge the robot has in the system in which it is *already* following T. In the terminology of [FHMV94], this lemma can be viewed as stating that an optimal termination condition is an implementation of the knowledge-based program P_g:

if $K_r g$ **then** halt

in the context κ corresponding to the uninhibited system $\mathcal{I}$. This is called a *knowledge-based* program because it contains tests that depend on the robot's knowledge. A *standard* program is one in which all tests depended on directly computable tests.

We could hope to use Lemma 3 in order to obtain an alternative characterization of optimal sound and complete termination conditions. Indeed, we can show:

Lemma 4 *Given G and κ, if the program P_g has a unique implementation in the context κ, then the condition defined by $K_r g$ in the system $\mathcal{I}_g$ representing P_g in the context κ is an optimal sound termination condition for $\mathcal{I}(\kappa)$ and G.*

Unfortunately, knowledge-based programs are not always guaranteed to have a unique (up to equivalence) implementation in terms of a standard program. There are examples of knowledge-based programs that can be implemented in inherently different ways in a given context. Nevertheless, most of these examples are somewhat unnatural, in some essential way making use of knowledge about the robot's future history to determine its current action. Fagin et al. [FHMV94] provide a sufficient condition for there being a unique standard implementation for a knowledge-based program P in a given context κ. The condition is stated as "the tests in P depend only on the past in the context κ". While the definition of this condition is somewhat technical, it applies in many applications of knowledge-based programs. Interestingly, even in fairly well-behaved motion planning contexts, the program P_g is not guaranteed to have a unique implementation.

Example 2 *Looking back on Example 1. In this case, $g = in[2,4]$. There are two implementations of the knowledge based protocol P_g in the context defined by this example. They are:*

1. **if** $r = 3$ **then** halt*; and*
2. **if** $r \geq 3$ **then** halt.

In both cases the local states obtained by translating $K_r(in[2,4])$ based on the systems obtained from termination conditions $T_1 : r = 3$, and $T_2 : r \geq 3$, are $r = 3$ and $r \geq 3$ respectively. Intuitively, in the system $\mathcal{I}(T_2)$ the robot "knows" it is following the condition T_2. The reasoning behind the condition T_2 could be viewed as the robot knowing that "if the condition T_2 has not been satisfied in the past, then the robot is in the goal". Notice that the condition $T_2 : r \geq 3$ guarantees that the robot cannot exit the goal region without halting, since its reading at location $q = 4$ must satisfy $r \geq 3$. Thus, although in the uninhibited system corresponding to this example a reading of, say, $r = 5$ does not ensure that the robot is in the goal region, in the system $\mathcal{I}(T_2)$ it does. We remark that the fact that the implementation of $\mathsf{P_g}$ in this context is not unique does not depend on the robot's movement being continuous. A discrete analogue of this example can be shown to display the exact same property.

In the discrete case, adding to the robot's local state a "local clock" whose value is guaranteed to increase every time the local state changes, is enough in order to ensure that the test for $K_r g$ that appears in P_g will "depend only on the past". It is an interesting open question whether adding a local clock in the continuous case would guarantee a unique implementation of P_g, and if so, what are the weakest properties one should require of the context and the clock for this to hold.

6 Conclusion and future work

Knowledge is a powerful tool for reasoning about domains in which uncertainty exists. The temporal-epistemic language we used provides a natural and powerful tool in the domain of motion planning with uncertainty, and enabled us to express and prove results more general than when using geometric specifications. One important task for future research will be to

look for interesting temporal/epistemic properties of different sensors and domains (relating the knowledge level and the geometric level), and exploit these properties to prove more specific results.

The present paper has studied two related problems in motion planning. However, knowledge can be applied to many additional natural problems, especially ones that deal with multiple agents, where purely geometrical reasoning would become even more complicated. In fact, motion planning offers all the problems encountered in distributed systems and more, but in a much richer setting. Another field of application involves tasks that have geometric as well as well as non-geometric aspects, where motion planning alone would not suffice. For example, parking a car in the goal area while making sure that other agent's are aware of its position. Knowledge's abstraction ability enables uniform treatment of such problems.

Sometimes we are willing to settle for less than knowledge. That is, we are willing to relax the soundness requirement and only demand that when we stop we are 'pretty sure' that we are in the goal. For this purpose, it seems likely that using some notion of belief, rather than knowledge, will be appropriate.

Acknowledgement We wish to thank Alvaro del Val, Nir Friedman, Joe Halpern, Ronny Kohavi and Moshe Tennenholtz for important discussions concerning this work.

Ronen Brafman and Yoav Shoham are supported in part by AFOSR grant F49620-92-J-0547; Jean-Claude Latombe is supported in part by ARPA grant N00014-92-J-1809; Yoram Moses is supported in part by a Helen and Milton A. Kimmelman Career Development Chair.

References

[BLS93] R. I. Brafman, J. C. Latombe, and Y. Shoham. Towards knowledge-level analysis of motion planning. In *Proceedings of the Eleventh National Conference on AI*, pages 670–675, 1993.

[Can89] J. F. Canny. On computability of fine motion plans. In *Proc. of the 1989 IEEE Int. Conf. on Robotics and Automation*, pages 177–182, 1989.

[DM90] C. Dwork and Y. Moses. Knowledge and common knowledge in a byzantine environment: crash failures. *Information and Computation*, 88(2):156–186, 1990.

[EH85] E. A. Emerson and J. Y. Halpern. Decision procedures and expressivenessin the temporal logic of branching time. *J. of Comp. and Sys. Sci.*, 30(1):1–24, 1985.

[Erd86] M. Erdmann. Using backprojection for fine motion planning with uncertainty. *Int. J. of Robotics Research*, 5(1):19–45, 1986.

[FHMV94] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge.* MIT Press, 1994. to appear.

[HF89] J. Y. Halpern and R. Fagin. Modelling knowledge and action in distributed systems. *Distributed Computing*, 3:159–177, 1989.

[Hin62] J. Hintikka. *Knowledge and Belief.* Cornell University Press, Ithaca, NY, 1962.

[HM90] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *J. ACM*, 37(3):549–587, 1990.

[KR90] L. P. Kaelbling and S. J. Rosenschein. Action and planning in embedded agents. *Robotics and Autonomous Systems*, 6:35–48, 1990.

[Lat91] J. C. Latombe. *Robot Motion Planning.* Kluwer Academinc Publishers, Boston, 1991.

[Lev84] H. Levesque. Foundations of a functional approach to knowledge representation. *Artificial Intelligence*, 23(2):155–212, 1984.

[LLS91] J. C. Latombe, A. Lazanas, and S. Shekhar. Robot motion planning with uncertainty in control and sensing. *Artificial Intelligence*, 52:1–47, 1991.

[LPMT84] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor. Automatic synthesis of fine-motion strategies for robots. *Int. J. of Robotics Research*, 3(1):3–24, 1984.

[Moo85] R. C. Moore. A formal theory of knowledge and action. In J. R. Hobbs and R. C. Moore, editors, *Formal Theories of the Common Sense World*, Norwood, N.J., 1985. Ablex Publishing Corporation.

[MT88] Y. Moses and M. R. Tuttle. Programming simultaneous actions using common knowledge. *Algorithmica*, 3:121–169, 1988.

[Ros85] S. J. Rosenschein. Formal theories of knowledge in ai and robotics. *New Generation Comp.*, 3:345–357, 1985.

[TMG88] R. H. Taylor, M. T. Mason, and K. Y. Goldberg. Sensor-based manipulation planning as a game with nature. In R. Bolles and B. Roth, editors, *Robotics Research 4*, pages 421–429. MIT Press, 1988.