On Comparing the Power of Robots

Jason M. O'Kane and Steven M. LaValle*

Abstract

Robots must complete their tasks in spite of unreliable actuators and limited, noisy sensing. In this paper, we consider the *information requirements* of such tasks. What sensing and actuation abilities are needed to complete a given task? Are some robot systems provably "more powerful," in terms of the tasks they can complete, than others? Can we find meaningful equivalence classes of robot systems? This line of research is inspired by the theory of computation, which has produced similar results for abstract computing machines. The basic idea is a dominance relation over robot systems that formalizes the idea that some robots are stronger than others. This comparison, which is based on the how the robots progress through their *information spaces*, induces a partial order over the set of robot systems. We prove some basic properties of this partial order and show that it is directly related to the robots' ability to complete tasks. We give examples to demonstrate the theory, including a detailed analysis of a limited-sensing global localization problem.

1 Introduction

Suppose we want a robot to complete some task, such as navigating to a goal, manipulating an object, or localizing itself within its environment. Many different combinations of sensing and motion modalities can be (and have been) used to complete each of these tasks. Indeed, much of the robotics literature is concerned with finding *sufficient conditions* on the sensing and actuation capabilities needed to complete such tasks. In this paper we take a different approach. For a given task, we are interested in determining the *necessary conditions*: What sensors and actuators are needed? What are the *information requirements* of robotic tasks? The long-term goal of this research is to develop a theory of robots and sensing that helps in answering such questions. Answers to these questions are important because we expect that a deep understanding of the difficulty of tasks in terms of their information requirements will lead to simpler and less expensive robot designs.

This work is inspired in part by the theory of computation, which begins with precisely defined models of abstract machines, such as finite automata, Turing machines, and so on [39]. In this context, a *problem* is usually a language of strings; to solve the problem is to accept strings in this language and reject all others. The theory of computation gives answers several kinds of basic questions about these machines and problems.

1. *Solvability*: Can a given machine can solve a given problem?
2. *Complexity*: If the machine can solve the problem, how efficiently (in terms of time or space, for example) can it do so?
3. *Comparison*: Are some machines strictly more powerful, in terms of the problems they can solve, than others? It is known, for example, that pushdown automata can accept a strictly larger set of languages than can finite automata. Likewise, Turing machines are more powerful than pushdown automata.

*This work is supported by ONR Grant N00014-02-1-0488, by DARPA grants #HR0011-05-1-0008 and #HR0011-07-1-0002. J. M. O'Kane (corresponding author) and S. M. LaValle are with the Department of Computer Science, University of Illinois at Urbana-Champaign, 201 North Goodwin Avenue, Urbana, IL 61801, USA. Email: {jokane, lavalle}@cs.uiuc.edu. Fax: +1-217-265-6591

4. *Equivalence*: Are there apparently dissimilar machines that can solve the same set of problems? For example, it is a standard result that a Turing machine with multiple tapes is functionally equivalent to an ordinary single-tape Turing machine. Less obviously, Turing machines and recursive functions have been shown to have equivalent computation power.

These ideas are well understood. In the sense that they form the formal foundation of the discipline, they are part of the core of computer science. Current robotic science lacks a comparable foundation; the field needs a unified theory in which meaningful statements can be made about the complexity of robotic tasks and the robot systems we build to complete these tasks.

Can we adapt standard models of computation to the robotics context? Unfortunately, these models are fundamentally ill-suited for studying robotics problems, because they assume that all of the relevant information is supplied ahead of time on the machine's tape. Sensing and uncertainty are central defining issues in robotics; this structure is destroyed by an *a priori* encoding of the problem on a machine's tape. Traditional models of online computation (see, for example, [18, 44, 73]) are also inadequate, because they assume that some fixed encoding of the problem is revealed incrementally. In contrast, robotics problems are generally interactive, in the sense that the robot's decisions influence the information that becomes available in the future. Others study robotics problems using similar tools [33, 67], but do not explicitly consider the effects of varying sensing and motion capabilities.

The aim of this paper is to develop a "sensor-centered" theory for analyzing and comparing robot systems. Our approach to doing so is based on two main ideas.

1. *Information spaces:* Traditional planning methods focus on the robot's progression through a space of states. What happens when the state is hidden and sensing thereby becomes relevant? One approach is to use *state estimation*, in which the robot uses the information available to it to make an "educated guess" about its state. The robot can treat this estimated state as its true state and ignore the uncertainty. In some extremely limited contexts this is provably optimal (see for example, Section 6.1 of [14]). We, however, are interested in a broader class of tasks for which accurate state estimation is impossible.

   The relevant space for such problems is the robot's *information space*. This space fully describes the information available to the robot, including its initial condition, the history of actions it has applied, and the history of sensor observations it has received. The robot's "state" in this space is always fully known. Information spaces originated in game theory [47], but have been used in robotics for some time [11, 28, 37, 50, 51].

2. *Tradeoffs expressed as partial orders:* We present a partial order defining the *dominance* of one robot system over another. The definition is based in turn on another partial order, an *information preference relation* over information space, that indicates which information states are "more informed" than others. Although these relations admit the possibility that no meaningful comparisons can be made, we find this desirable: physical tasks and robot systems exhibit complex relationships and tradeoffs that can potentially defy meaningful linear ordering.

The central idea we present in this paper is a notion of *dominance* of one robot model over another. In informal terms:

> A robot R_2 *dominates* another robot R_1 if R_2 can "simulate" R_1, collecting at least as much information as R_1.

We make three primary contributions in developing this idea. First, we present the idea of *robotic primitives* for modeling robot systems as collections of independent components. A single robotic primitive represents a self-contained "instruction set" for the robot that may involve sensing, motion, or both. A robot model is defined by a set of primitives that the robot can use to complete its task. By selecting a "catalog" of primitives from which complete robot systems are constructed, we effectively determine a set of robot systems to consider. For clarity, we define these models in an idealized setting in which time is modelled as a series of discrete stages and the robot has perfect knowledge of its environment, perfect control and perfect sensing.

Second, we give a definition for dominance of one robot system over another that formalizes the imprecise definition above. This definition is based on comparing reachability in a *derived information space* [50]. By mapping sensor-action histories from a variety of robots into the same derived information space, we can compare the abilities of these robots in a concrete, formal way. We prove some basic properties of this dominance relation and give some examples, including a detailed investigation of the global localization problem. Third, we demonstrate the generality of our ideas by showing how to remove several of the simplifying assumptions we make in the initial presentation.

The remainder of this paper is organized as follows. Section 2 reviews related research. Section 3 lays a foundation of basic definitions for robotic planning problems. Section 4 introduces the concept of robotic primitives and defines the set of robots induced by a catalog of primitives. In Section 5 we describe the information preference relation, a partial ordering over derived information space that formalizes the idea that some information states are better than others. The definition of dominance and some basic properties thereof appear in Section 6. In Section 7, we apply the results from Sections 4-6 to the global localization task. In Section 8 we present several generalizations our basic results to account for environment uncertainty, imperfect control and sensing, and continuous time. Section 9 contains discussion and conclusions.

Preliminary versions of this work appear in [62] and [63].

2 Related work

Our approach can be viewed as *minimalist* in the sense that we are interested in solutions that use sensing sparingly. The minimalist approach in robotics has a long history, dating perhaps to Whitney [79]. Minimalist approaches have been used in manufacturing contexts for part orientation [4, 5, 30, 36, 37, 57, 77, 80] and in mobile robotics for navigation and exploration [3, 21, 42, 48, 54, 59, 76].

Our goals are similar to those of Donald [25]. The reductions in that work are similar to our dominance relation; Donald's notion of calibration is related to our idea of initial conditions. The most fundamental difference is that our analysis is rooted in the information space. We claim that for robotic problems for which sensing is a crucial issue, the information space is the space in which the problem can most naturally be posed. The work of Erdmann [29] is grounded in the preimage planning ideas due to Lozano-Perez, Mason, and Taylor [53]. In Erdmann's work, sensors are modeled by giving a partition of state space. The problem of sensor design is to choose a partition so that from each region in the partition, the robot knows what action to select in order to make progress toward its goal. Others in artificial intelligence [19] and control theory [1, 27, 34] have addressed related issues.

Although the examples in this paper use nondeterministic uncertainty, which is based on set membership, the basic structure of our analysis is compatible with probabilistic uncertainty models like those of [75]. Many probabilistic methods (for example, [7, 52]) can be characterized as operating in an information space whose members are probability distributions over state space. Our methods can be viewed as axiomatic because they can be applied in any situation that satisfies the definitions of Sections 3-5. In this sense, the model of uncertainty used is orthogonal to the questions addressed in this work.

3 Basic definitions

This section presents basic definitions for robotic planning problems. In order to keep the presentation as clear as possible, we make several simplifying assumptions here and show in Section 8 how to relax them.

3.1 States, actions, and observations

We allow a robot to move in a state space X. Many of the examples in this paper are for a point robot with orientation in the plane. In these examples, we use $X = E \times S^1$, in which $E \subset \mathbb{R}^2$ is the robot's environment and $S^1 = [0, 2\pi]/\sim$, where $\sim$ is an equivalence relation identifying 0 and 2π, represents the robot's orientation. In general, however, we allow arbitrary state spaces, including configuration spaces and

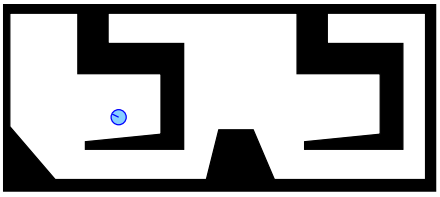

Figure 1: A robot in a planar environment E. Its state space is $X = E \times S^1$.

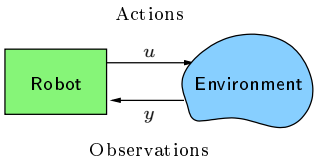

Figure 2: The robot interacts with its environment by executing actions and receiving observations.

phase spaces of physical systems. Section 8.1 considers the particular case that arises when a robot moves amidst unknown obstacles.

Time proceeds in variable-length *stages*, indexed by consecutive integers starting with 1. In each stage, the robot selects an action u from its action space U and moves to a new state according a state transition function $f : X \times U \to X$. At the conclusion of each stage, the robot's sensors provide an observation y from an observation space Y, according to $h : X \times U \to Y$. Call h the robot's *observation function*. Let x_k, u_k, and y_k denote respectively the state, action, and observation at stage k. These sequences are related to each other by f and h:

$$x_{k+1} = f(x_k, u_k) \tag{1}$$

$$y_k = h(x_k, y_k). \tag{2}$$

Although we are assuming in this section that both state transitions and observations are deterministic, we acknowledge that in realistic contexts, managing unpredictability in motion and sensing is a crucial issue. We omit such uncertainty here because of the additional complications it would introduce. The extensions needed to relax this assumption are introduced in Section 8.

For convenience, we also define an iterated version of f that applies k actions in succession:

$$f^k(x, u_1, \ldots, u_k) = f(\cdots f(f(x, u_1), u_2) \cdots), u_k). \tag{3}$$

The robot's capabilities are modeled in the action and observation sets U and Y and in the maps f and h that interpret these sets. See Fig. 2. A *robot model* is a 5-tuple (X, U, Y, f, h) giving values to each of these elements.

3.2 Information spaces

Although the robot does not know its state, it does have access to the history of actions it has selected and observations it has made. The space of such histories is the robot's *history information space* (history I-space), denoted $\mathcal{I}_{hist}$:

$$\mathcal{I}_{hist} = \bigcup_{i=0}^{\infty} (U \times Y)^i. \tag{4}$$

After k stages, the robot's *history information state* (history I-state) is a sequence of length $2k$: $\eta = (u_1, y_1, \ldots, u_k, y_k)$. We occasionally abuse notation by writing (η, u_{k+1}, y_{k+1}) for the history I-state formed

by appending u_{k+1} and y_{k+1} to η. How is the state space related to the robot's history I-space? One connection is by way to the notion of states consistent with an I-state:

Definition 1 *A state $x \in X$ is* consistent with *a history I-state $\eta = (u_1, y_1, \ldots, u_k, y_k)$ if there exists some $x_1 \in X$ such that $x = f^k(x_1, u_1, \ldots, u_k)$ and $y_j = h(f^{j-1}(x_1, u_1, \ldots, u_{j-1}), u_j)$ for each $j = 1, \ldots, k$.* ∘

The intuition is that a state x is consistent with an I-state η if a robot having I-state η might possibly be at state x.

We may define a *policy* $\pi : \mathcal{I}_{hist} \to U$ over history I-space. As a shorthand, we recursively define a function F that applies a policy several times in succession, starting with some state x:

$$F^0(\eta, \pi, x) = \eta \tag{5}$$

$$F^k(\eta, \pi, x) = (\eta_{k-1}, \pi(F^{k-1}(\eta, \pi, x)), h(x, \pi(F^{k-1}(\eta, \pi, x)))), \tag{6}$$

Note that $F^k(\eta, \pi, x)$ depends on the true state x (which is unknown to the robot) because x influences the observation sequence that the robot receives.

The history I-space is not particularly useful by itself. For pairs of robots whose action or observation spaces differ, the history I-spaces also differ, making the history I-space unhelpful for comparing robots. For these reasons, we select a *derived information space* (derived I-space) $\mathcal{I}$ and an *information mapping* (I-map) $\kappa : \mathcal{I}_{hist} \to \mathcal{I}$. Informally, an I-map computes a "compression" or "interpretation" of the history I-state. If the history I-spaces of several robot models are mapped to the same derived I-space $\mathcal{I}$, then the robots can be compared by examining their progression through $\mathcal{I}$. In principle, we may select $\mathcal{I}$ and κ arbitrarily. The usefulness of a derived I-space lies in its ability to fully capture the information relevant to the task of interest.

Example 1 *We define the* nondeterministic I-space $\mathcal{I}_{ndet}$*, in which derived I-states are nonempty subsets of X. The interpretation is that the robot's derived I-state is the minimal set guaranteed to contain the true state. For any history I-state η, the nondeterministic derived I-state $\kappa_{ndet}(\eta)$ is the set of states consistent with η. Equivalently, the I-map $\kappa_{ndet} : \mathcal{I}_{hist} \to \mathcal{I}_{ndet}$ can be defined recursively:*

$$\kappa_{ndet}(\,) = X \tag{7}$$

$$\kappa_{ndet}(\eta, u, y) = \{f(x, u) \mid x \in \kappa_{ndet}(\eta), y = h(x, u)\} \tag{8}$$

Note that in Equation 7, we assume the robot initially has no information about its state. ◇

An important special case is the value of κ for an empty history, that is, $\kappa(\,)$. This value gives an *initial condition* for the robot, reflecting any knowledge the robot may have before its execution begins.

A *task* for the robot is a goal region $\mathcal{I}_G \subseteq \mathcal{I}$ in information space that the robot must reach. This notion is a generalization of the traditional idea of a goal state or goal region in state space. A *solution* is a policy π under which, for any $x \in X$, there exists l such that $F^l(\eta_1, \pi, x) \in \mathcal{I}_G$.

4 Defining a set of robot systems

In this section we discuss how a set of robots can be defined in terms of a set of independent components.

4.1 Robotic primitives

At the most concrete level, a robot is a collection of motors and sensors connected to some sort of computer. Between these components there may be interactions via open- or closed-loop controls. We abstract this complexity by defining the notion of a *robotic primitive*. Each robotic primitive defines a "mode of operation" for the robot. When primitives are implemented, they may draw on one or more of the robot's physical

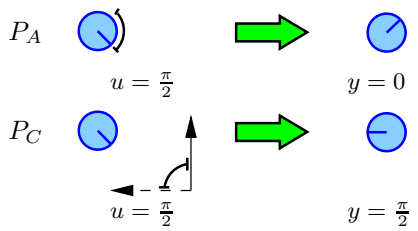

Figure 3: Sample executions of the primitives of Examples 2 and 3. [top] P_A allows the robot rotate relative to its current orientation. [bottom] P_C allows the robot to rotate relative to a globally defined "north" direction.

sensors or actuators. Every kind of motion or sensing available to the robot must be modeled as a robotic primitive. Robotic primitives correspond to the *oracles* that occur in the theory of computation [72], in the sense that they provide the ability to make certain transitions and collect certain observations, without specifying how these abilities are implemented.

Formally, we define robotic primitives in terms of the action and observation abilities they provide.

Definition 2 *A* robotic primitive *(or simply a* primitive*) is a 4-tuple*

$$(U_i, Y_i, f_i, h_i) \tag{9}$$

giving an action set U_i, an observation set Y_i, a state transition function $f_i : X \times U_i \to X$, and an observation function $h_i : X \times U_i \to Y_i$. ◦

We now give several examples to illustrate the idea. Examples 3-7 apply to a point robot with orientation in the plane, so $X = \mathbb{R}^2 \times S^1$. Illustrations of these primitives appear in Figures 3-5. We revisit these examples in Sections 6 and 7.

Example 2 *Let $P_A = (S^1, \{0\}, f_A, h_A)$. Let f_A compute relative rotations, so that from a state $x = (x_1, x_2, \theta)$, we have $f_A(x, u) = (x_1, x_2, \theta + u)$. Since $Y_A = \{0\}$ contains only a dummy element, h_A is a trivial function always returning 0. This primitive can be implemented with an angular odometer on a mobile robot capable of rotating in place.* ◇

Example 3 *Let $P_C = (S^1 \sqcup \{0\}, S^1, f_C, h_C)$. The $\sqcup$ notation indicates a disjoint union operation, under which identical elements from different source sets remain distinct. Define $f_C(x, u)$ to set the rotation coordinate of x to equal u if $u \in S^1$ or to leave x unchanged if $u \in \{0\}$. The observation function h_C returns the robot's final orientation. This primitive amounts to allowing the robot to orient itself with respect to a global reference frame, or to sense its current orientation without rotating. This primitive can be implemented using a compass on a robot that can rotate in place.* ◇

Example 4 *Let $P_T = (\{0\}, \{0\}, f_T, h_T)$. Define f_T to compute a forward translation to the obstacle boundary. This primitive can be implemented with a contact sensor on a mobile robot that can reliably move forward.* ◇

Example 5 *Let $P_L = ([0, \infty), [0, \infty), f_L, h_L)$. For $x \in X$ and $u \in U$, define $f_L(x, u)$ to compute a forward translation of distance at most u, stopping short only if the robot reaches an obstacle first. The observation $h_L(x, u)$ is the actual distance traveled. This primitive can be implemented with a linear odometer. Depending on implementation issues, a contact sensor may be needed as well.* ◇

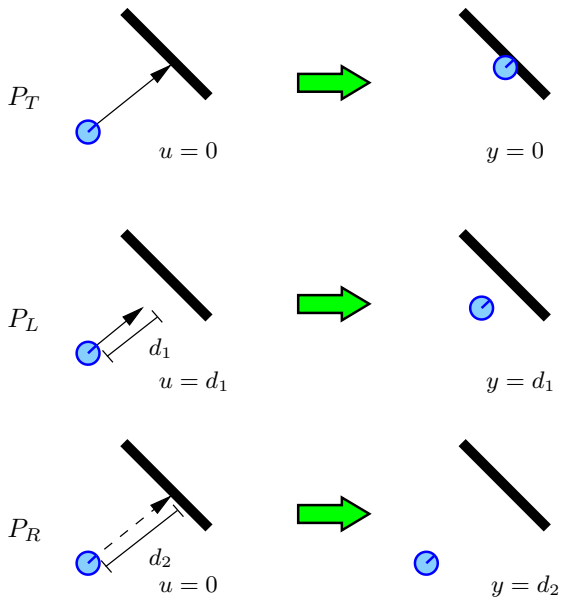

Figure 4: Sample executions of the primitives of Examples 4-6. [top] P_T allows the robot to translate forward until it reaches an obstacle. [middle] P_L allows a robot to specify a distance to translate. [bottom] P_R allows the robot to measure the distance forward to the nearest obstacle, but does not change the robot's state.

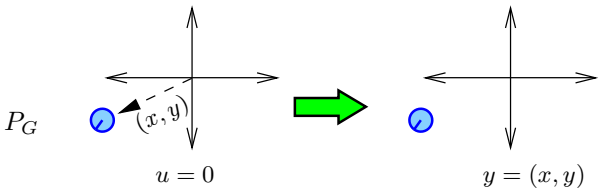

Figure 5: A sample execution of the primitive of Example 7. The robot senses its position, but its state does not change.

Example 6 *Let $P_R = (\{0\}, [0, \infty), f_R, h_R)$. Again, $f(x, u) = x$ for all x and u. The observation $h(x, u)$ is the distance to the nearest obstacle directly in front of the robot. This primitive models the capabilities of a forward-facing unidirectional range sensor.* ◇

Example 7 *Let $P_G = (\{0\}, \mathbb{R}^2, f_G, h_G)$. For all $x \in X$, $f_G(x, 0) = x$, so that this primitive never changes the robot's state. For a state $x = (x_1, x_2, \theta)$, let $h(x, 0) = (x_1, x_2)$. This primitive roughly corresponds to a GPS device that the robot can periodically poll to determine its location in the plane.* ◇

Other possibilities for primitives include landmark detectors, wall followers, visibility sensors, and so on. A more complete listing of sensors suitable for adaptation into robotic primitives appears in Section 11.5.1 of [50].

We claim that there are several benefits to modeling robot systems as collections of primitives. First, we claim that robotic primitives represent the right level of abstraction at which *planning* problems are interesting but manageable. If we consider sensors at too fine a level of detail, the problem takes on the

character of a closed-loop control system. If the primitives are too sophisticated, we risk trivializing the planning problem while creating an unbearable modeling burden. Second, by dividing time into stages, we avoid the technical difficulties of describing the robot's progression through $\mathcal{I}$ in continuous time. This consideration is increasingly important if we allow noise to affect state transitions or observations. We address issues related to the modeling of time more completely in Section 8.3.

Let $\mathcal{RP} = \{P_A, \ldots, P_N\}$ denote a catalog of primitives. We may form a robot model by selecting nonempty subset of $\mathcal{RP}$. A robot defined by the primitive set $R = \{P_{i_1}, \ldots, P_{i_m}\} \subseteq \mathcal{RP}$ has action set $U_R = U_{i_1} \sqcup \cdots \sqcup U_{i_m}$ and observation set $Y_R = Y_{i_1} \sqcup \cdots \sqcup Y_{i_m}$. The state transition function $f_R : X \times U_R \to X$, and observation function $h_R : X \times U_R \to Y_R$, are formed by unioning the f and h maps from the relevant primitives. When it can be done without ambiguity, we sometimes the use phrase *robot model* to refer directly to the set of primitives, rather than to the 5-tuple (X, U, Y, f, h) formed by these primitives. With this usage, it is meaningful to apply set operations such as union or intersection directly to robots.

5 The information preference relation

Our goal is a dominance relation under which we can declare one robot "better than" another. To do so, we need a formal notion of one I-state being "more informed" than another. To that end, choose a derived I-space $\mathcal{I}$ and an I-map κ into $\mathcal{I}$. Equip $\mathcal{I}$ with a partial order, which we call an *information preference relation*. Write $\kappa(\eta_1) \preceq \kappa(\eta_2)$ to indicate that $\kappa(\eta_2)$ is a refinement of $\kappa(\eta_1)$. We require that for any $\eta_1, \eta_2 \in \mathcal{I}_{hist}$, and for any $u \in U$ and $y \in Y$,

$$\kappa(\eta_1) \preceq \kappa(\eta_2) \implies \kappa(\eta_1, u, y) \preceq \kappa(\eta_2, u, y). \tag{10}$$

This is a consistency property requiring preference for one I-state over another to be preserved across transitions in I-space.

Example 8 *Regardless of $\mathcal{I}$ or κ, it is well-defined (but perhaps unhelpful) to use a trivial relation under which $\kappa(\eta_1) \preceq \kappa(\eta_2)$ if and only if $\kappa(\eta_1) = \kappa(\eta_2)$.* ◇

Example 9 *Under nondeterministic uncertainty, we can define $\kappa_{ndet}(\eta_1) \preceq \kappa_{ndet}(\eta_2)$ if and only if $\kappa_{ndet}(\eta_2) \subseteq \kappa_{ndet}(\eta_1)$. To show that (10) is satisfied, suppose $\kappa_{ndet}(\eta_1) \preceq \kappa_{ndet}(\eta_2)$. Let $x \in \kappa_{ndet}(\eta_2, u, y)$. The definition of κ_{ndet} ensures that there exists some $x' \in \kappa_{ndet}(\eta_2)$ such that $f(x', u) = x$ and $h(x', u) = y$. However, because $\kappa_{ndet}(\eta_2) \subseteq \kappa_{ndet}(\eta_1)$, we have $x' \in \kappa_{ndet}(\eta_1)$. It follows that $x \in \kappa_{ndet}(\eta_1, u, y)$.* ◇

The information preference relation we choose affects the goal regions that are sensible to consider. We should select a region in which, for every I-state in the region, we also include any I-states preferable to it. Definition 3 codifies this idea of a sensible goal region.

Definition 3 *Consider a set $I \subset \mathcal{I}$ of derived I-states. If, for any $\eta_1 \in I$ and $\eta_2 \in \mathcal{I}$ with $\eta_1 \preceq \eta_2$, we have $\eta_2 \in I$, then I is* preference closed. ◦

Alternatively, we can view preference closure as a constraint on $\preceq$. Fixing a space $\mathcal{G}$ of potential goal regions, we admit a partial order $\preceq$ only if every region in $\mathcal{G}$ is preference closed under $\preceq$. The trivial definition of $\preceq$ in Example 8 always passes this test, regardless of $\mathcal{G}$.

6 A dominance relation over robot systems

Now we turn our attention to a definition of dominance of one robot system over another. This dominance relation induces a partial order over robot systems according to their sensing and actuation abilities. The intuition is that dominance is based on one robot's ability to "simulate" another.

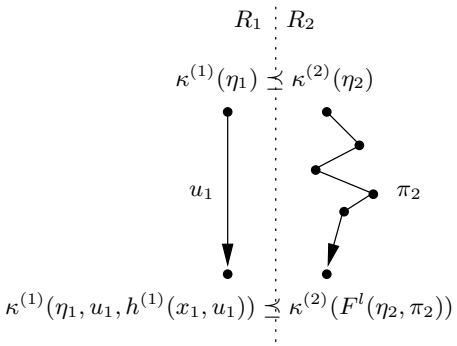

Figure 6: An illustration of Definition 4. If R_2 can always reach an I-state better than the one reached by R_1, then $R_1 \trianglelefteq R_2$.

Definition 4 *[Robot dominance] Consider two robots* $R^{(1)} = (X^{(1)}, U^{(1)}, Y^{(1)}, f^{(1)}, h^{(1)})$ *and* $R^{(2)} = (X^{(2)}, U^{(2)}, Y^{(2)}, f^{(2)}, h^{(2)})$. *Choose a derived I-space* $\mathcal{I}$ *and I-maps* $\kappa^{(1)} : X^{(1)} \to \mathcal{I}$ *and* $\kappa^{(2)} : X^{(2)} \to \mathcal{I}$. *If, for all*

- $\eta_1 \in \mathcal{I}^{(1)}_{hist}$,
- $\eta_2 \in \mathcal{I}^{(2)}_{hist}$ *for which* $\kappa^{(1)}(\eta_1) \preceq \kappa^{(2)}(\eta_2)$, *and all*
- $u_1 \in U^{(1)}$,

there exists a policy $\pi_2 : \mathcal{I}^{(2)}_{hist} \to U^{(2)}$ *such that for all* $x_1 \in X^{(1)}$ *consistent with* η_1 *and all* $x_2 \in X^{(2)}$ *consistent with* η_2, *there exists and a positive integer* l *such that*

$$\kappa^{(1)}(\eta_1, u_1, h^{(1)}(x_1, u_1)) \preceq \kappa(F^l(\eta_2, \pi_2, x_2)), \tag{11}$$

then R_2 dominates R_1 *under* $\mathcal{I}$ *and* κ, *denoted* $R_1 \trianglelefteq R_2$. *If* $R_1 \trianglelefteq R_2$ *and* $R_2 \trianglelefteq R_1$, *then* R_1 *and* R_2 *are* equivalent, *denoted* $R_1 \equiv R_2$. *If* $R_1 \not\trianglelefteq R_2$ *and* $R_2 \not\trianglelefteq R_1$ *then* R_1 *and* R_2 *are* incomparable, *denoted* $R_1 \not\trianglelefteq R_2$. ∘

Informally, Definition 4 means that, for any transition made by R_1, there exists some strategy for R_2 to reach an information state at least as good, in the sense of information preference, as that reached by R_1. This is what we mean when we describe the statement $R_1 \trianglelefteq R_2$ as meaning that R_2 can simulate R_1. See Fig. 6.

6.1 Dominance examples

Several examples will clarify the definition.

Example 10 *Let* $R_1 = \{P_R\}$ *and* $R_2 = \{P_A, P_L\}$. *Recall the definitions of these primitives from Examples 3, 5, and 6. We argue under nondeterministic uncertainty that* $R_1 \trianglelefteq R_2$ *by showing that* R_2 *can simulate* R_1 *in the precise sense of Definition 4. Let* $\eta_1 \in \mathcal{I}^{(1)}_{hist}$ *and* $\eta_2 \in \mathcal{I}^{(2)}_{hist}$ *with* $\kappa(\eta_1) \preceq \kappa(\eta_2)$. *Since* $U_1 = \{0\}$, *there is only one choice for* u_1. *Let* $l = 4$ *and define* π_2 *so that* R_2, *starting from* η_2, *executes these actions in succession:*

(1) Use P_L *with a very large input to move forward to the nearest obstacle. Let* $d = h(x, u)$ *denote the distance moved.*

(2) Use P_A *with* $u = 180^\circ$ *to perform a half turn.*

(3) Use P_L *with* $u = d$ *to return the robot to its initial position.*

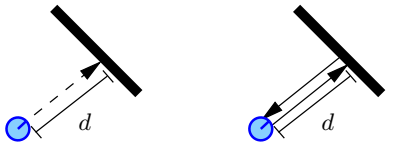

Figure 7: An illustration of Example 10. The robot $R_2 = \{P_A, P_L\}$ dominates the robot $R_1 = \{P_R\}$ because the former can simulate the latter. [left] A distance measurement made directly by R_1. [right] Distance is measured indirectly by R_2 using its linear odometer.

(4) Use P_A with $u = 180^\circ$ to perform a half turn, returning the robot to its original orientation.

This policy is illustrated in Figure 7. It is easy to verify that from any $x \in X$, we have

$$\kappa(\eta_1, u_1, h(x, u_1)) \preceq \kappa(F^4(\eta_2, \pi_2, x)),$$

and therefore $R_1 \trianglelefteq R_2$. Since R_1, which is completely immobile, cannot simulate the translations or rotations of R_2, we have $R_2 \not\trianglelefteq R_1$.

Note that these relationships are based on the robots' ability to move through $\mathcal{I}_{ndet}$, and do not consider any notion of the cost of motion or sensing. The introduction of such a cost function would likely lead to Pareto optima that express tradeoffs between the complexity of sensing built into the robot and the execution costs of particular plans executed by the robot. We do not consider such tradeoffs here. ◇

Example 11 *Let $R_1 = \{P_T\}$ and $R_2 = \{P_L\}$. We show under nondeterministic uncertainty that $R_1 \trianglelefteq R_2$. Let $\eta_1 \in \mathcal{I}^{(1)}_{hist}$ and $\eta_2 \in \mathcal{I}^{(2)}_{hist}$ with $\eta_1 \preceq \eta_2$. There is only one choice for u_1. Choose $l = 1$ and define π_2 to choose an input for P_L larger than the diameter of the environment. This causes the motions of R_1 and R_2 to be identical. The resulting derived I-states η_1' and η_2' for R_1 and R_2 are the same, except that R_2 receives a meaningful sensor reading that may cause one or more candidates to be pruned. This sensor information only makes η_2' smaller, so the preference $\eta_1' \preceq \eta_2'$ is preserved. Conclude that $R_1 \trianglelefteq R_2$.* ◇

It bears emphasis that the relation induced by Definition 4 depends on the I-maps used. The next two examples illustrate this.

Example 12 *Let $R_1 = \{P_A\}$ and $R_2 = \{P_C\}$. We argue that $R_1 \trianglelefteq R_2$ under the usual nondeterministic I-map with the initial condition of total uncertainty. Let $\eta_1 \in \mathcal{I}^{(1)}_{hist}$ and $\eta_2 \in \mathcal{I}^{(2)}_{hist}$ with $\eta_1 \preceq \eta_2$. Let $u_1 \in U_1 = S^1$. Choose $l = 2$ and define π_2 to select the following two actions:*

(1) Use P_C with $u = 0$ to sense the robot's orientation without changing the state. Let θ denote this orientation.

(2) Use P_C to rotate the robot to orientation $\theta + u$ in the global frame.

As in Example 11, the resulting states for R_1 and R_2 are identical but, since R_2 knows its orientation, it may be able to eliminate some candidates that R_1 cannot. This establishes that $R_1 \trianglelefteq R_2$. Are R_1 and R_2 equivalent under this I-map? No, because R_2 can, with a single action, sense its orientation, but this information can never be gathered by R_1. Therefore $R_2 \not\trianglelefteq R_1$ and $R_1 \not\equiv R_2$. ◇

Example 13 *Consider a situation identical to that of Example 12, but modify κ for a different initial condition $\kappa(\,) = \mathbb{R}^2 \times \{\pi/2\}$. That is, the robot begins its execution knowing its initial orientation. At every step, R_1 knows its orientation in the global frame, and can simulate R_2 using angle addition. Therefore we have $R_2 \trianglelefteq R_1$. But using the same reasoning as in Example 12, we know $R_1 \trianglelefteq R_2$. Therefore, for this κ, we have $R_1 \equiv R_2$.* ◇

6.2 Properties of the dominance relation

We conclude this section with some basic properties that follow from Definition 4.

Lemma 1 *The dominance relation $\trianglelefteq$ is partial order. Likewise $\equiv$ is indeed an equivalence relation.*

Lemma 2 *For any three robots R_1, R_2 and R_3 for which $R_1 \trianglelefteq R_2$:*

(a) $R_1 \trianglelefteq R_1 \cup R_3$ (Adding primitives never hurts)

(b) $R_2 \equiv R_2 \cup R_1$ (Redundancy doesn't help)

(c) $R_1 \cup R_3 \trianglelefteq R_2 \cup R_3$ (No unexpected interactions)

Proof: (a) Let $\eta_1 \in \mathcal{I}^{(1)}_{hist}$, $\eta_{13} \in \mathcal{I}^{(13)}_{hist}$, and $u_1 \in U_1$. Assume $\kappa(\eta_1) \preceq \kappa(\eta_{13})$. Choose $l = 1$ and $\pi_{13}(\eta) = u_1$ for all η. We have $\kappa(\eta_1, u_1, h(x, u_1)) \preceq \kappa(\eta_{13}, u_1, h(x, u_1)) = \kappa(F^l(\eta_{13}, \pi_{13}, x))$, completing the proof.

(b) Since $R_2 \cup R_2 = R_2$, it follows from part (a) that $R_2 \trianglelefteq R_1 \cup R_2$. It remains to show that $R_2 \cup R_1 \trianglelefteq R_2$. Let $\eta_2 \in \mathcal{I}^{(2)}_{hist}$, $\eta_{12} \in \mathcal{I}^{(12)}_{hist}$, and $u_2 \in U_2$. Assume $\kappa(\eta_2) \preceq \kappa(\eta_{12})$. Choose $l = 1$ and $\pi_{12}(\eta) = u_2$ for all η. We have $\kappa(\eta_B, u_B, h(x, u_B)) \preceq \kappa(\eta_{12}, u_B, h(x, u_B)) = \kappa(F^l(\eta_{12}, \pi_{12}, x))$, completing the proof.

(c) Let $\eta_{13} \in \mathcal{I}^{(13)}_{hist}$, $\eta_{23} \in \mathcal{I}^{(23)}_{hist}$, and $u_{13} \in U_1 \sqcup U_3$. Assume $\kappa(\eta_{13}) \preceq \kappa(\eta_{23})$. Either $u_{13} \in U_1$ or $u_{13} \in U_3$. If $u_{13} \in U_1$, then because $R_1 \trianglelefteq R_2$ there exist π_3 and l satisfying the definition for $R_1 \cup R_3 \trianglelefteq R_2 \cup R_3$. If $u_{13} \in U_3$, then choose $l = 1$ and $\pi_{23}(\eta) = u_{13}$ for all η. For all x, we have $\kappa(\eta_{13}, u_{13}, h(x, u_{13})) \preceq \kappa(\eta_{23}, u_{13}, h(x, u_{13})) = \kappa(F^l(\eta_{23}, \pi_{23}, x))$, completing the proof. □

Corollary 3 *If $R_1 \equiv R_2$, then $R_1 \cup R_3 \equiv R_2 \cup R_3$.*

Proof: Apply Lemma 2c twice. □

Lemma 2c might be misleading. Certainly, hardware components can be made to interact in interesting ways. For example, a control system might combine information from linear and angular odometers to execute circular arc motions. This apparent contradiction results from the definition of robotic primitives, which execute *serially*, rather than in parallel. In this sense, robotic primitives model complete "packages" of sensing and actuation strategies, rather than the individual sensors or motors themselves.

Lastly, we connect the idea of dominance to the ability of robots to complete tasks.

Lemma 4 (Solution by imitation) *Consider two robots R_1 and R_2 with $R_1 \trianglelefteq R_2$ and a preference-closed goal region $\mathcal{I}_G$. If R_1 can reach $\mathcal{I}_G$ then R_2 can reach $\mathcal{I}_G$.*

Proof: Use the policy π_2 implied by Definition 4 to complete the task with R_2. □

7 Extended example: Global localization

In this section we present a detailed example using the definitions of Sections 5 and 6. We consider a *global localization* task, in which the robot has an accurate map of its environment but has no knowledge of its position within that environment. Many forms of the localization problem with varying sensing modalities have been studied in great detail. Some methods [8, 12, 22–24, 38, 49, 74, 78] passively observe the motions of the robot in order to draw conclusions about the robot's state. Others [26, 45, 46, 60, 61, 68, 69] actively drive the robot to reduce uncertainty. The purpose of this example is to show how the results of Section 6 can be used to discover the information requirements of this particular problem in robotics. An analogy can be made to the classification of languages in the theory of computation. It has been shown, for example, that to accept the language of palindromes requires a machine with computation abilities at least as powerful as a pushdown automaton. In this section, we derive similar results regarding the sensing and motion abilities needed to complete the active global localization task.

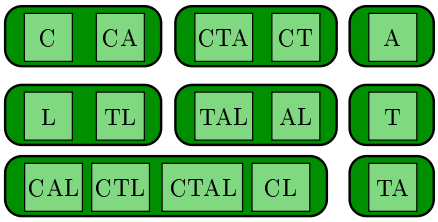

Figure 8: Fifteen robot models grouped into their eight equivalence classes.

7.1 Task definition

Let $E \subseteq \mathbb{R}^2$ denote a planar environment in which a point robot moves. Assume that E is polygonal, bounded, closed, and simply-connected and that the rotational symmetry group of E is trivial. The robot's state space is $X = E \times S^1$, accounting for its position within E and its orientation. We consider a catalog $\mathcal{RP} = \{P_A, P_C, P_T, P_L\}$ of four primitives from Examples 2-4. From these primitives we can form 15 distinct robots. For brevity, we use concatenation to indicate the primitives with which a robot is equipped, so that CT refers to a robot with primitive set $\{P_C, P_T\}$; similar names apply to the other 14 robot models.

Select $\mathcal{I} = \text{pow}(X) - \emptyset$. For κ, use the nondeterministic map defined in Example 1. The initial condition is total uncertainty, so $\kappa(\,) = X$. For the information preference relation, use the definition from Example 9, in which information preference is defined by subset containment. The goal region for the localization task is

$$\mathcal{I}_G = \{\eta \in \mathcal{I} \mid |\eta| = 1\}. \tag{12}$$

That is, we want to command the robot so that only a single final state is consistent with its history I-state. If the robot can complete the task for any E consistent with the assumptions above, we say that the robot can localize itself.

7.2 Equivalences and dominances

Although $\mathcal{RP}$ generates 15 robot models, we can use the results of Section 6 to group them into equivalence classes.

Lemma 5 *The following equivalences hold:*

(a) $CA \equiv C$

(b) $CTA \equiv CT$

(c) $TL \equiv L$

(d) $TAL \equiv AL$

(e) $CAL \equiv CTL \equiv CTAL \equiv CL$

The three remaining robot models, A, T, and AT, are in singleton equivalence classes.

Proof: (a) Combine Example 12 and Lemma 2b. (b) Combine Example 12, Lemma 2b, and Corollary 3. (c) Combine Example 11 and Lemma 2b. (d) Combine Example 11, Lemma 2b, and Corollary 3. (e) Combine Examples 11 and 12, Lemma 2b, and Corollary 3. □

These equivalences are illustrated in Figure 8. From each, select the unique robot with the fewest primitives and discard the remaining 7 robots. We can state a number of dominances between these classes.

Lemma 6 *Between representatives of the equivalence classes from Lemma 5, the following dominances hold:*

(a) $C \trianglelefteq CT \trianglelefteq CL$

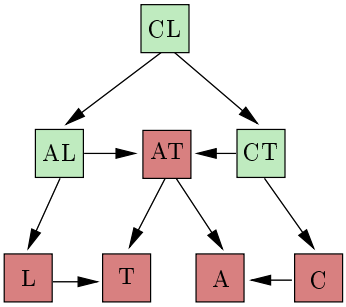

Figure 9: Classification of robot models under which the localization task can be completed. Shaded models do not admit a solution. Arrows indicate dominances.

(b) $A \trianglelefteq AT \trianglelefteq AL \trianglelefteq CL$

(c) $L \trianglelefteq AL \trianglelefteq CL$

(d) $T \trianglelefteq AT \trianglelefteq CT \trianglelefteq CL$

Proof: Combine Examples 11 and 12 with Lemma 2a. □

7.3 Completing the localization task

Which equivalence classes contain robots that can complete the localization task? First, notice that several robot models are so absurdly simple that we can rule them out immediately.

Lemma 7 *None of C, A, L, and T can localize themselves.*

Proof: For C and A, notice that no action changes the robot's position and no observation is influenced by position. Therefore neither robot can ever gather information about its position. For L and T, notice that the robot can never change its orientation. Information available to the robot is limited to the ray extending from its initial state to the nearest obstacle forward. Since E may contain continua of starting states consistent with this information, neither robot can localize itself. □

Prior results are helpful for the remaining cases.

Lemma 8 ([64]) *AL and CT can localize themselves but AT cannot.*

Finally, we can finish the classification. The results of Lemmas 7-9 are summarized in Figure 9.

Lemma 9 *CT can localize itself.*

Proof: Combine Lemma 4 with Lemma 8. □

The result is a complete description of the power of the primitives in $\mathcal{RP}$, in terms of dominance and equivalence, and a complete classification of the solvability of the localization problem over this hierarchy.

8 Extensions and generalizations

This section contains a series of extensions and generalizations to the techniques presented in Sections 3-6. Our intention is to illustrate that, although the preceding results are for a class of highly idealized systems, the general structure of our analysis is useful for a wider variety of problems with greater degrees of realism and generality. We propose methods for dealing with unknown environments (Section 8.1), with sensing and control uncertainty (Section 8.2), and with continuous time (Section 8.3). Although we present each method separately, the extensions are orthogonal in the sense that it its straightforward to apply all of them at once.

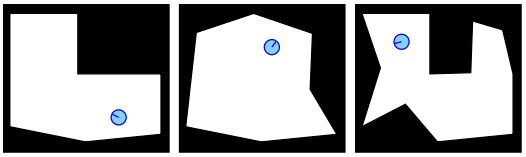

Figure 10: Three states for an example system containing a mobile robot in the plane with environment uncertainty. When the environment is uncertain, the identity of the environment becomes part of the state of the system.

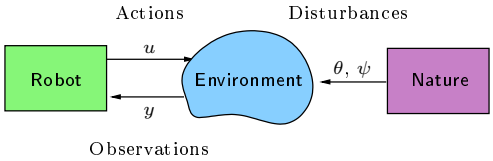

Figure 11: As the robot interacts with its environment, an artificial decision maker nature generates disturbances.

8.1 Unknown environments

In the preceding analysis, we tacitly assumed that the robot moves in a fixed, known environment. What happens when the robot begins with limited or no knowledge about its environment, in the sense that positions and geometry of obstacles, map topology, navigability of terrain, and so on are unknown? Imperfect knowledge about the environment is a more drastic instance of the general issue of state uncertainty. If the state is defined to include a description of the environment in addition to the robot's configuration, then uncertainty in the environment can be represented as an additional dimension of state uncertainty.

Concretely, choose an *environment space* $\mathcal{E}$ of which each element $E \in \mathcal{E}$ is a potential environment for the robot. Possibilities for $\mathcal{E}$ with varying degrees of realism, interest, practicality, and amenability to analysis, include:

1. the set of bounded planar grids with occupancy maps,
2. the set of simple polygons in the plane, and
3. the set of compact regions in $\mathbb{R}^2$ or $\mathbb{R}^3$ with connected interiors and piecewise analytic boundaries.
4. the set of terrain maps from $\mathbb{R}^2$ to $\mathbb{R}$, giving the elevation or navigability at each point in the plane.

The state space is formed by combining the robot's configuration space $\mathcal{C}$ with $\mathcal{E}$, so that $X = \mathcal{C} \times \mathcal{E}$. In the complete model, the true environment $E \in \mathcal{E}$ affects the robot by influencing the state transitions that the robot makes and the observations that the robot receives. Since the only change is to use a more complicated state space, Definition 4 need not change, and the results of Section 6 still hold.

8.2 Imperfect sensing and control

We have assumed so far that the robot can execute all of its actions with perfect precision and complete reliability. The motions of real robots are imprecise and unpredictable. Moreover, although we have accounted for the importance of sensing by assuming that the robot is uncertain of its current state and must rely on sensing, we have assumed that sensor readings are uncorrupted by noise. A more realistic sensor model would allow information from sensors to be subject to error.

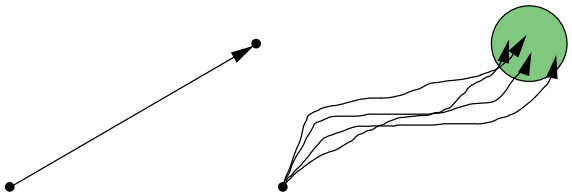

Figure 12: [left] The robot in Example 14 gives displacement inputs that determine a nominal trajectory. [right] Nature interferes with this motion, but error bounds ensure that the final state is contained in a circle of radius $k\theta_{max}$.

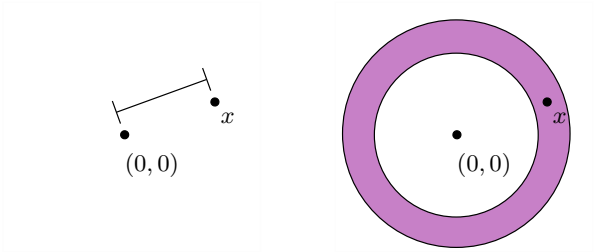

Figure 13: [left] The robot in Example 15 has a sensor that reports a noisy estimate of the distance to the origin. [right] Accounting for noise bounded by ψ_{max}, the observation confines the robot's state to an annulus of width $2\psi_{max}$.

We propose to follow the approach used in game theory [15, 66] and represent this uncertainty by envisioning an abstract external decision maker called "nature." The current state, the action chosen by the robot, and the choices made by nature combine to determine how the state changes; given this information, the state trajectory is fully determined. Formally, define a *nature action space* Θ and augment the state transition function f to depend on nature's choice of $\theta \in \Theta$, so that $f : X \times U \times \Theta \to X$. Nature affects the robot's observations in a similar way. Define a *nature observation action space* Ψ and redefine the observation function $h : X \times U \times \Psi \to Y$.

Example 14 *Consider a point robot that can move freely in the plane by issuing displacement commands, but whose motion is subject to noise. Let u_{max} denote a bound on the magnitude of the displacement in each stage, and let θ_{max} denote a bound on magnitude of the error in this displacement. Let $X = \mathbb{R}^2$, $U = \{u \in \mathbb{R}^2 \mid ||u|| \leq u_{max}\}$, $\Theta = \{\theta \in \mathbb{R}^2 \mid ||\theta|| \leq \theta_{max}\}$, and $f(x, u, \theta) = x + u + \theta$. At stage k, the robot can be certain that its state lies within a closed ball of radius $k\theta_{max}$, centered at the nominal (error free) final point. See Figure 12.* ◇

Example 15 *Suppose a mobile robot has a sensor that detects the distance to some landmark. Let $X = \mathbb{R}^2$ and $Y = \mathbb{R}$. Without loss of generality, position the landmark at the origin. Assume that the sensor has bounded additive error, so that $\Psi = [-\psi_{max}, \psi_{max}]$ and $h(x, \psi) = ||x|| + \psi$. See Figure 13. At each stage, the robot knows that its state is within an annulus of width $2\psi_{max}$ centered at the origin.* ◇

In the presence of interference from nature, there are at least two relevant solution concepts.

1. A strategy $\pi : \mathcal{I}_{hist} \to U$ is a *possible solution* if there exists some stage k and choices of $\theta_1, \ldots, \theta_k$ and $\psi_1, \ldots, \psi_k$ for which the robot reaches an information state $\eta_k \in \mathcal{I}_G$. The robot may reach $\mathcal{I}_G$, but it is also possible that control or sensing errors will prevent it from achieving this goal.

2. A strategy $\pi : \mathcal{I}_{hist} \to U$ is a *guaranteed solution* if there exists some stage k such that for all choices of $\theta_1, \ldots, \theta_k$ and $\psi_1, \ldots, \psi_k$, the robot reaches an information state $\eta_k \in \mathcal{I}_G$. The robot can always reach its goal, regardless of any interference by nature.

Other solution concepts, such as those based on performance bounds or on probabilistic guarantees of reaching the goal, are possible but we do not consider them here. In this context, Definition 4 must be generalized to include universal quantifiers over nature's actions.

Definition 5 *[Robot dominance with sensing and control error] Consider two robot systems* $R^{(1)} = (X^{(1)}, U^{(1)}, Y^{(1)}, \Theta^{(1)}, \Psi^{(1)}, f^{(1)}, h^{(1)})$ *and* $R^{(2)} = (X^{(2)}, U^{(2)}, Y^{(2)}, \Theta^{(2)}, \Psi^{(2)}, f^{(2)}, h^{(2)})$*. Choose a derived I-space* $\mathcal{I}$ *and I-maps* $\kappa^{(1)} : X^{(1)} \to \mathcal{I}$ *and* $\kappa^{(2)} : X^{(2)} \to \mathcal{I}$*. If, for all*

- $\eta_1 \in \mathcal{I}^{(1)}_{hist}$,
- $\eta_2 \in \mathcal{I}^{(2)}_{hist}$ *for which* $\kappa^{(1)}(\eta_1) \preceq \kappa^{(2)}(\eta_2)$*, and all*
- $u_1 \in U^{(1)}$,

there exists a policy $\pi_2 : \mathcal{I}^{(2)}_{hist} \to U^{(2)}$ *such that for all* $x_1 \in X^{(1)}$ *consistent with* η_1 *and all* $x_2 \in X^{(2)}$ *consistent with* η_2*, there exists a positive integer* l *such that for all*

- $\theta_1 \in \Theta^{(1)}$,
- $\psi_1 \in \Psi^{(1)}$,
- $\theta_{2,1}, \ldots, \theta_{2,l} \in \Theta^{(2)}$,
- $\psi_{2,1}, \ldots, \psi_{2,l} \in \psi^{(2)}$,

we have

$$\kappa^{(1)}(\eta_1, u_1, h^{(1)}(x_1, u_1)) \preceq \kappa(F^l(\eta_2, \pi_2, x_2)) \tag{13}$$

then R_2 dominates R_1 *under* $\mathcal{I}$ *and* κ*, denoted* $R_1 \trianglelefteq R_2$*. If* $R_1 \trianglelefteq R_2$ *and* $R_2 \trianglelefteq R_1$*, then* R_1 *and* R_2 *are* equivalent*, denoted* $R_1 \equiv R_2$*.* ∘

The next example demonstrates that Definition 5 behaves reasonably.

Example 16 (Varying error bounds) *Recall the incompletely specified models in Examples 14 and 15. Consider two robot systems* R_1 *and* R_2 *with state transitions as in Example 14 and observations as in Example 15;* R_1 *and* R_2 *differ only in the error bounds* $\theta^{(1)}_{max}$*,* $\psi^{(1)}_{max}$*,* $\theta^{(2)}_{max}$*, and* $\psi^{(2)}_{max}$*. We compare these robots under* κ_{ndet}*. Comparing* $\theta^{(1)}_{max}$ *to* $\theta^{(2)}_{max}$*, and* $\psi^{(1)}_{max}$ *to* $\psi^{(2)}_{max}$*, there are four cases:*

1. *If* $\theta^{(1)}_{max} = \theta^{(2)}_{max}$ *and* $\psi^{(1)}_{max} = \psi^{(2)}_{max}$*, then* $R_1 \equiv R_2$*.*
2. *If* $\theta^{(1)}_{max} \leq \theta^{(2)}_{max}$ *and* $\psi^{(1)}_{max} \leq \psi^{(2)}_{max}$*, then* $R_2 \trianglelefteq R_1$*.*
3. *If* $\theta^{(2)}_{max} \leq \theta^{(1)}_{max}$ *and* $\psi^{(2)}_{max} \leq \psi^{(1)}_{max}$*, then* $R_1 \trianglelefteq R_2$*.*
4. *Otherwise,* $R_2 \bowtie R_1$*.*

These results follow in a straightforward manner from Definition 5. The intuition is that one robot system dominates the other if and only if its error bounds are smaller. ◇

8.3 Continuous time

The models presented to this point manage time in discrete stages, in which the robot makes a single decision at each stage. This discretization of time may be unsatisfactory for many kinds of systems, especially those that require complicated control strategies. Continuous-time models have a more direct correspondence with

reality. To make the appropriate generalizations, we must replace the discrete sequences of states, actions, and observations with functions of a continuous time parameter t.

The state space X, action space U, and observation space Y remain unchanged from the discrete stage formulation. At each instant t, the robot chooses some $u(t) \in U$. Let $\widetilde{U}_t$ denote the space of all functions from $[0, t)$ into U, and let $\widetilde{U} = \bigcup_{t \in [0,\infty)} \widetilde{U}_t$. For simplicity of notation, adopt the convention that $[0, 0) = \emptyset$. Define $\widetilde{u} : [0, \infty) \to U$ as the robot's complete action history, and let $\widetilde{u}_t \in \widetilde{U}$ denote the robot's action history up to (but exclusive of) time t. We include a special *termination action* $u_T \in U$. The robot selects u_T to indicate that it has finished its task and intends to terminate execution. We require that if $u(t) = u_T$, then $u(t') = u_T$ for all $t' > t$. We describe changes in the state with a *state transition function*

$$\Phi : X \times \bigcup_{t \in [0,\infty)} \widetilde{U}_t \to X. \tag{14}$$

The intuition is that, given a starting state $x(0)$, and an action history $\widetilde{u}_t$, the state transition function computes the resulting state

$$x(t) = \Phi(x(0), \widetilde{u}_t). \tag{15}$$

This notation of a "black box" state transition function follows notation employed in control theory, for example by Chen [20].

Example 17 *A familiar special case of (15) occurs if $\widetilde{u}$ is a smooth function and there exists a function f such that*

$$\Phi(x(0), \widetilde{u}_t) = x(0) + \int_0^t f(x(s), u(s)) \mathrm{d}s. \tag{16}$$

In this case, the system dynamics can be described by the differential equation $\dot{x} = f(x, u)$. ◇

As time passes, the robot's sensors provide feedback in the form of observations drawn from an observation space Y. Let $\widetilde{Y}_t$ denote the space of functions mapping $[0, t]$ into Y and let $\widetilde{Y} = \bigcup_{t \in [0,\infty)} \widetilde{Y}_t$. The robot's complete observation history is $\widetilde{y} : [0, \infty) \to Y$. The observation history up to t (inclusive) is $\widetilde{y}_t \in \widetilde{Y}_t$. The observations received by the robot are governed by the observation function[1] $h : X \to Y$. The history I-state becomes

$$\mathcal{I}_{hist} = \bigcup_{t \in [0,\infty)} \widetilde{U}_t \times \widetilde{Y}_t, \tag{17}$$

and the history I-state at time t is $\eta(t) = (\widetilde{u}_t, \widetilde{y}_t) \in \mathcal{I}_{hist}$. A state x is consistent with an I-state $\eta(t) = (\widetilde{u}_t, \widetilde{y}_t)$ if and only if there exists some starting state $x(0)$ such that $\Phi(x(0), \widetilde{u}_t) = x$ and $h(x(t')) = y(t')$ for $t' < t$.

Strategies for the robot are given as feedback plans over $\mathcal{I}_{hist}$. We describe the robot's strategy as a feedback strategy $\pi : \mathcal{I}_{hist} \to U$ that specifies an action for history I-state. The next two examples illustrate that feedback over a derived I-space can be a natural way to express familiar kinds of strategies.

Example 18 (Open loop strategy) *Let $\mathcal{I}_{time} = [0, \infty)$ and consider the I-map $\kappa_{time}(\eta(t)) = t$. In this case, the derived I-state is simply the time elapsed. If the robot has an intended open loop action trajectory $\omega : [0, t_f) \to U$, a strategy to execute γ is $\pi(\eta(t)) = \omega(\kappa_{time}(\eta(t)))$ if $t < t_f$ and $\pi(\eta(t)) = u_T$ otherwise.* ◇

[1] In our discrete-stage formulation, we used a slightly different observation model, in which $h : X \times U \to Y$. In a continuous-time adaptation, the time period over which observations are available is the half-open interval $[0, t)$; $\widetilde{y}_t$ would be undefined at t itself. As a result, the closest we could come to a memoryless strategy is to use the left-hand limit of $\widetilde{y}_t$ at t, $\kappa_{obs}(\eta(t)) = \lim_{t' \to t^-} y(t')$, provided the limit exists. (Compare to Example 19.) This technicality is part of the motivation for preventing y from depending directly on u, as we have done in this section. A more complete treatment of these kinds of sensor models appears in Section 11.1.1 of [50].

Example 19 (Memoryless strategy) *Another possibility is that it is enough to know the "most recent" observation, so $\mathcal{I}_{obs} = Y$ and $\kappa_{obs}(\eta(t)) = y(t)$. Given a memoryless plan $\gamma : Y \to U$, the composed function $\kappa_{obs} \circ \gamma : \mathcal{I}_{hist} \to U$ is a memoryless information feedback strategy.* ◇

We assume that a given strategy is executed until it selects u_T. The time when this occurs, the resulting final state, and the observations received along the way are all affected by the strategy π itself and the starting state $x(0)$. Assuming that the robot executes π, the termination time is

$$T(\pi, x(0)) = \inf\{t \in [0, \infty) \mid \pi(\eta(t)) = u_T\}, \tag{18}$$

and the final state is $F(\pi, x(0)) = \Phi(x(0), \widetilde{u}_{T(\pi,x(0))})$.

Example 20 (Concatenating strategies) *Given two strategies π_1 and π_2, a new strategy that concatenates them (that is, executes them in sequence) is expressed by $\pi(\eta(t)) = \pi_1(\eta(t))$ if $\pi_1(\eta(t)) \neq u_T$ and $\pi(\eta(t)) = \pi_2(\eta(t))$ otherwise. By nesting this construction, arbitrarily many strategies can be chained together.* ◇

Definition 4 generalizes in a natural way.

Definition 6 *[Robot dominance in continuous time] Consider two continuous-time robot systems $R_1 = (X^{(1)}, U^{(1)}, Y^{(1)}, \Phi^{(1)}, h^{(1)})$, and $R_2 = (X^{(2)}, U^{(2)}, Y^{(2)}, \Phi^{(2)}, h^{(2)})$. If, for all*

- *$\eta^{(1)}(t_1) \in \mathcal{I}^{(1)}_{hist}$,*
- *$\eta^{(2)}(t_2) \in \mathcal{I}^{(2)}_{hist}$ for which $\kappa^{(1)}(\eta^{(1)}(t_1)) \preceq \kappa^{(2)}(\eta^{(2)}(t_2))$,*
- *$t'_1 \in [0, \infty)$, and all*
- *$\widetilde{u}^{(1)}_{t'_1} \in \widetilde{U}^{(1)}_{t'_1}$,*

there exists an information feedback strategy $\pi_2 : \mathcal{I}^{(2)}_{hist} \to U^{(2)}$, such that for all $x^{(1)} \in X^{(1)}$ consistent with $\eta^{(1)}(t_1)$ and $x^{(2)} \in X^{(2)}$ consistent with $\eta^{(2)}(t_2)$, there exists $t'_2 \in [0, \infty)$ such that if R_1 executes $\widetilde{u}^{(1)}_{t'_1}$ from time t_1 to t'_1 and R_2 executes $\pi^{(2)}$ from time t_2 to t'_2, we have

$$\kappa^{(1)}(\eta^{(1)}(t'_1)) \preceq \kappa^{(2)}(\eta^{(2)}(t'_2)) \tag{19}$$

then R_2 dominates *R_1, denoted $R_1 \trianglelefteq R_2$.* ◦

See Figure 14. The next two examples illustrate the implications of Definition 6.

Example 21 (Omniscient sensing and perfect control) *Consider a degenerate case with $Y = X$, and $h(x) = x$. This situation gives the robot complete information about its state. Choose $\kappa(\eta(t)) = y(t) = x(t)$. Let $\eta_1 \preceq \eta_2$ if and only if $\eta_1 = \eta_2$. In this context, Definition 4 becomes a statement about the regions of state space reachable by different control systems.*

Suppose three such systems R_1, R_2, and R_3 differ only in their action spaces $U^{(1)}$, $U^{(2)}$, and $U^{(3)}$. Let $Z(A)$ denote the subset of state space reachable by a robot with action space A. Suppose $R_1 \trianglelefteq R_2$. R_3 need not be comparable to either R_1 or R_2. Note that additional robot models can be constructed from unions of $U^{(1)}$, $U^{(2)}$, and $U^{(3)}$. We have the following results:

$$Z(U^{(1)}) \subseteq Z(U^{(2)} \cup U^{(3)}) \tag{20}$$

$$Z(U^{(1)}) = Z(U^{(1)} \cup U^{(2)}) \tag{21}$$

$$Z(U^{(1)} \cup U^{(3)}) \subseteq Z(U^{(2)} \cup U^{(3)}) \tag{22}$$

These results are analogous to Lemma 2. Note that in combining action spaces in this way, we allow the robot to choose sequentially *the action set from which to choose its action. The results fail if the robot is somehow allowed to choose actions from each constituent set in parallel.* ◇

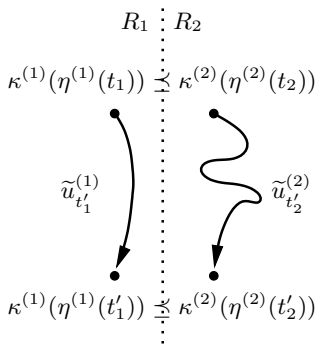

Figure 14: An illustration of Definition 6. Compare to figure 6.

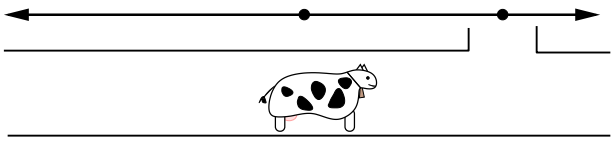

Figure 15: The lost cow of Example 22 searching for a gate.

Example 22 (A Lost Cow) *A well-known problem in online algorithms is the* lost cow problem *[10,43] in which a near-sighted cow moves along a fence searching for a gate, as illustrated in Figure 15. The difficulty under the standard sensing model is that the cow must systematically search in both directions from its initial position without any information about the distance or direction to the gate. The interest in this problem derives from potential applications in (or at least the potential for better understanding of) exploration in unbounded environments.*

We formulate the lost cow problem and consider how the sensing model affects the cow's searching ability. Let $X = \mathbb{R}$, in which $x(t)$ is the position of the gate relative to the cow at time t. Let the action space be $U = [-1, 1]$ with $\Phi(x(0), \widetilde{u}_t) = x(0) + \int_0^t u(s)\mathrm{d}s$. We compare three distinct models C_1, C_2, and C_3 under κ_{ndet}.

1. *C_1: Let $Y^{(1)} = \mathbb{R}$ and $h^{(1)}(x) = x$. Here the cow can determine both the direction and distance to the gate.*
2. *C_2: Let $Y^{(2)} = \{-1, 0, 1\}$ and $h(x) = \mathrm{sign}(x)$. This allows the cow to determine the direction it must move to reach the gate, but not the distance.*
3. *C_3: Let $Y^{(3)} = \{0, 1\}$ and $h^{(2)}(x) = 1$ if $x = 0$ and $h^{(2)}(x) = 1$ otherwise. This is the standard lost cow sensing model, in which the cow cannot see the gate from a distance, but can detect the gate when it arrives.*

Perhaps surprisingly, these three models are equivalent in the sense of Definition 6. This is because each can eventually determine its state (by finding the gate) and after the state is known, state uncertainty cannot recur. To simulate C_1 with C_3, first execute the algorithm of [10], then move to the state occupied by C_1. $\diamond$

We conclude our discussion of continuous-time models by showing how a discrete stage model in the form of Section 3 can be constructed from a continuous-time model in the form presented above. Consider a division of time into variable length stages, in which, in each stage, the robot executes a single information feedback strategy to completion. We require of each of these strategies the following special property:

Definition 7 *[History invariance] If, for all $\eta(t) \in \mathcal{I}_{hist}$, all $x \in X$ consistent with $\eta(t)$, and all $y(0) \in Y$, we have $F(\pi, x, \eta(t)) = F(\pi, x, \eta(0))$, then π is a* history-invariant *strategy.* ∘

The intuition of the definition is that the robot executing π is free to use the observation and action history generated during its own execution, but it cannot peer into the past before its execution began in order to make decisions. Given a continuous-time robot system $R = (X, U, Y, \Phi, h)$ (as defined in this section) and a set Π of history-invariant information feedback strategies, construct a discrete-stage system (as in Section 3) $\overline{R} = (X, \overline{U}, \overline{Y}, \overline{f}, \overline{h})$ as follows:

1. The state space X is the same.
2. The action space is $\overline{U} = \Pi$.
3. The observation space is $\overline{Y} = \widetilde{Y}$.
4. The state transition function is $f : X \times \overline{U} \to X$, with $f(x, \pi) = F(\pi, x, \eta(0))$.
5. The observation function is $h : X \times \overline{U} \to \overline{Y}$.

The system starts at some (unknown) initial state $x_1 \in X$. Let $x_k \in X$, $u_k \in \overline{U}$, and $y_k \in \overline{Y}$, denote the appropriate values at stage k. These sequences are related to each other by $x_{k+1} = f(x_k, u_k)$ and $y_k = h(x_k, u_k)$. The history I-state consists of the action and observation histories: $\eta_k = (u_1, y_1, \ldots, u_{K-1}, y_{K-1})$. This construction gives a discrete-stage system faithful to the dynamics in both state space and I-space of the underlying continuous time system.

Lemma 10 *Any action sequence $u_1, \ldots, u_K$ executed by $\overline{R}$ reaches the same final state x and the analogous final history I-state as does R.*

Note, however, that in making this transformation, we must choose a set Π of strategies and may therefore restrict the space of plans that the robot can execute. If Π does not contain a sufficiently rich selection of information feedback strategies, there may be regions of I-space that are no longer reachable under the discretized model. In this way, Π is analogous to the catalog of robotic primitives $\mathcal{RP}$ introduced in Section 4.

9 Discussion

The results of this paper are intended to lay a foundation for a sensor-centered theory for comparing robotic problems and systems. Great potential exists to build on this foundation, particularly by pushing the analogy to the theory of computation even further.

The most obvious avenue for future work is to study a broader collection of problems. Although this paper considers an active global localization problem in detail, other fundamental robotics problems warrant similar analysis of their information requirements. For example, results exist for limited-sensing versions of navigation [42, 42, 54, 55, 67], exploration [2, 21, 59, 76], and manipulation [4–6, 30, 36] tasks. Using the techniques we have presented, it should be possible to unify and extend these results to develop a more complete understanding of the sensing and motion abilities needed to solve these problems. Other problems and more complex sensing systems could also be investigated.

One of the most powerful ideas in the theory of computation that we have not explored here is the idea of *reductions*, which hold promise for comparing robotic problems themselves. The resulting statements would have the form "Problem A is at least as hard as Problem B." To make things more concrete, we might consider *decision problems*, in which the robot must determine if its environment $E \in \mathcal{E}$ has a certain property. Such problems can be expressed naturally as planning problems in I-space. To decide if E has a property $\Xi : \mathcal{E} \to \{0, 1\}$, the robot must reach the goal region

$$\mathcal{I}_{G,\Xi} = \{\eta \in \mathcal{I}_{hist} \mid \forall (q, E) \in \kappa_{ndet}(\eta), \Xi(E) = 1\} \cup \{\eta \in \mathcal{I}_{hist} \mid \forall (q, E) \in \kappa_{ndet}(\eta), \Xi(E) = 0\}. \quad (23)$$

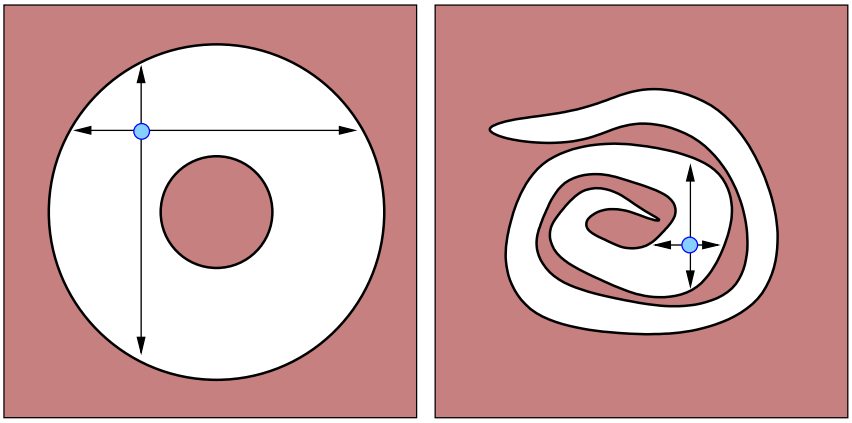

Figure 16: A sample decision problem. What sensing is required to decide if a planar environment is simply connected? What robots can distinguish the annulus environment on the left from the helix on the right?

An example is in Figure 16.

Another direction is to study the computational complexity of robotics problems. We expect that there exist rich tradeoffs between computation time, memory usage, sensing requirements, and solution quality. Some research has been done on computation requirements for certain tasks, for example [13, 17, 41], but very little is known in general. These computational issues must be approached with care, especially if those computations involve real numbers [16]. One way of approaching such issues is to study *sufficient I-maps* [50], which are I-maps for which transitions can be computed directly in the derived I-space, allowing the history to be discarded. For example, if a problem can be solved under a given robot model using a sufficient I-map into a finite derived I-space of cardinality n, the memory required to solve the problem is $O(\log n)$. The results of Blum and Kozen [17], for example, can be characterized as showing how a discrete exploration problem can be solved in a derived I-space with cardinality linear in the height of the area to be explored, meaning that only logarithmic memory is required. Similarly, the complexity of computing transitions in a derived I-space gives an indication of the computation time required to solve a given problem.

In spite of these possibilities, there are important limitations to the the idealized models we presented. Of the many issues remaining to be addressed, we mention a few here.

Probabilistic uncertainty We have focused our attention on nondeterministic uncertainty, but a large subset of contemporary work in robotics uses probabilistic models of uncertainty [7, 31, 40, 70, 71, 75]. Our results also apply, at least in principle, to probabilistic uncertainty. In this context, the relevant derived I-space is a space of probability distributions over X. It is not immediately clear what the "right" information preference relation over such a space would be. Depending on the models used, it may also be necessary to relax Definition 4 to require only that R_2 can simulate R_1 with sufficiently high probability. More generally, the differences between nondeterministic and probabilistic uncertainty models warrant further exploration. For example, nondeterministic uncertainty has the property that sensing can only help – actions from primitives like P_R (Example 6) or P_G (Example 7) that do not change the state always lead to a derived I-state at least as good as the current one. Under probabilistic uncertainty, this property does not hold; sensing can sometimes increase uncertainty.

Selecting the catalog of primitives Although we believe that our robotic primitives provide a useful abstraction, any results derived using our methods are meaningful only if $\mathcal{RP}$ is diverse enough to faithful represent the underlying system. It remains an open problem to systematically find small (or at least succinctly described) sets of robotic primitives that are complete (or nearly complete) in the sense of not

eliminating any reachable regions in I-space. There is, however, active interest in related problems for control systems [32, 35, 56, 58].

What happens if $\mathcal{RP}$ is not a finite set? For example, we may extend P_L (from Example 5) to a family $\{P_{L,\epsilon} = (S^1, \{0\}, f_{L_\epsilon}, h_{L_\epsilon}) \mid \epsilon \geq 0\}$ of primitives, each using a noisy linear odometer whose error is bounded by ϵ. If $\mathcal{RP}$ contains many such families of primitives, and we assume each robot has at most one primitive from each family, then the space of robot models is a cube in $\mathbb{R}^n$. The problem of identifying the region in which a given task can be solved is correspondingly more difficult.

Parameterization of time In Section 8.3, we parameterized the robot's observations by time. In doing so, we implicitly assumed that the robot has an accurate clock. Although such an assumption is generally not technologically impractical, it requires care in abstract models to ensure that the robot cannot acquire extra information "for free." A robot might, for example, use this implicit clock to parlay an accurate velocity sensor into a perfect odometer. One solution is to express $\widetilde{u}$ and $\widetilde{y}$ as functions of some other abstract parameter p. Then, to recover the original functions of time, the robot must determine a hidden mapping from $\mathbb{R}$ to $\mathbb{R}$ under which p maps to t.

Cooperation and coordination In this work we consider only a single independent robot. We might also consider the performance of teams of cooperative robots on the same tasks. Such work would require an investigation of the joint I-spaces that would arise from the interaction of multiple agents, each having only limited information. In particular, limited and possibly noisy communication between robots must be modeled. Many of the relevant issues are worked out in the game theory literature [9, 65].

References

[1] A. Abate, A. D. Ames, and S. Sastry, "Error-bounds based stochastic approximations and simulations of hybrid dynamical systems," in *American Control Conference*, 2006.

[2] E. U. Acar and H. Choset, "Complete sensor-based coverage with extended-range detectors: A hierarchical decomposition in terms of critical points and voronoi diagrams," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[3] ——, "Robust sensor-based coverage of unstructured environments," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[4] P. K. Agarwal, A. D. Collins, and J. L. Harer, "Minimal trap design," in *Proc. IEEE International Conference on Robotics and Automation*, vol. 3, 2001, pp. 2243–2248.

[5] S. Akella, W. Huang, K. M. Lynch, and M. T. Mason, "Parts feeding on a conveyor with a one joint robot," *Algorithmica*, vol. 26(3), pp. 313–344, March-April 2000.

[6] S. Akella and M. Mason, "Posing polygonal objects in the plane by pushing," *International Journal of Robotics Research*, vol. 17, no. 1, pp. 70–88, Jan. 1998.

[7] D. J. Austin and P. Jensfelt, "Using multiple gaussian hypotheses to represent probability distributions for mobile robot localization," in *Proc. IEEE International Conference on Robotics and Automation*, 2000, pp. 1036–1041.

[8] D. Avis and H. Imai, "Locating a robot with angle measurements," *J. Symb. Comput.*, vol. 10, no. 3-4, pp. 311–326, 1990.

[9] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory, 2nd Ed.* London: Academic, 1995.

[10] R. A. Baeza-Yates, J. C. Culberson, and G. J. E. Rawlins, "Searching in the plane," *Information and Computation*, vol. 106, pp. 234–252, 1993.

[11] J. Barraquand and P. Ferbach, "Motion planning with uncertainty: The information space approach," in *Proc. IEEE International Conference on Robotics and Automation*, 1995, pp. 1341–1348.

[12] K. Basye and T. Dean, "Map learning with indistinguishable locations," in *Proc. Conference on Uncertainty in Artificial Intelligence*. North-Holland, 1990, pp. 331–342.

[13] M. A. Bender, A. Fernández, D. Ron, A. Sahai, and S. Vadhan, "The power of a pebble: exploring and mapping directed graphs," in *Proc. IEEE Symposium on Foundations of Computer Science*, 1998, pp. 269–278.

[14] D. P. Bertsekas, *Dynamic programming and optimal control*, 2nd ed. Belmont, MA: Athena Scientic, 2001, vol. 1.

[15] D. Blackwell and M. A. Girshik, *Theory of Games and Statisitical Decisions*. New York: Dover, 1979.

[16] L. Blum, F. Cucker, M. Shub, and S. Smale, *Complexity and Real Computation*. Berlin: Springer Verlag, 1998.

[17] M. Blum and D. Kozen, "On the power of the compass (or, why mazes are easier to search than graphs)," in *Proc. IEEE Symposium on Foundations of Computer Science*, 1978, pp. 132–142.

[18] A. Borodin and R. El-Yaniv, *Online Computation and Competitive Analysis*. Cambridge, UK: Cambridge University Press, 98.

[19] R. I. Brafman, J. Y. Halpern, and Y. Shoham, "On the knowledge requirements of tasks," *Artificial Intelligence*, vol. 98, no. 1-2, pp. 317–349, 1998.

[20] C.-T. Chen, *Linear System Theory and Design*. New York: Holt, Rinehart, and Winston, 1984.

[21] H. Choset and J. Burdick, "Sensor based planning, part I: The generalized Voronoi graph," in *Proc. IEEE International Conference on Robotics and Automation*, 1995, pp. 1649–1655.

[22] I. J. Cox, "Blanche – an experiment in guidance and navigation of an autonomous robot vehicle," *IEEE Transations on Robotics and Automation*, vol. 7:2, pp. 193–204, 1991.

[23] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Proc. IEEE International Conference on Robotics and Automation*, 1999.

[24] E. D. Demaine, A. López-Ortiz, and J. I. Munro, "Robot localization without depth perception," in *Scandinavian Workshop on Algorithm Theory*, 2002.

[25] B. R. Donald, "On information invariants in robotics," *Artificial Intelligence*, vol. 72, no. 1-2, pp. 217–304, 1995.

[26] G. Dudek, K. Romanik, and S. Whitesides, "Localizing a robot with minimum travel," *SIAM J. Comput.*, vol. 27, no. 2, pp. 583–604, 1998.

[27] M. Egerstedt, "Motion description languages for multi-modal control in robotics," in *Control Problems in Robotics*, ser. Springer Tracts in Advanced Robotics, A. Bicchi, H. Cristensen, and D. Prattichizzo, Eds. Springer-Verlag, 2002, pp. 75–90.

[28] M. Erdmann, "Randomization for robot tasks: Using dynamic programming in the space of knowledge states," *Algorithmica*, vol. 10, pp. 248–291, 1993.

[29] ——, "Understanding action and sensing by designing action-based sensors," *International Journal of Robotics Research*, vol. 14, no. 5, 1995.

[30] M. Erdmann and M. T. Mason, "An exploration of sensorless manipulation," *IEEE Transations on Robotics and Automation*, vol. 4, no. 4, pp. 369–379, Aug. 1988.

[31] D. Fox, W. Burgard, and S. Thrun, "Active markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, pp. 195–207, 1998.

[32] E. Frazzoli, "Robust hybrid control of autonomous vehicle motion planning," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, June 2001.

[33] Y. Gabriely and E. Rimon, "Competitive complexity of mobile robot on line motion planning problems," in *Proc. Workshop on the Algorithmic Foundations of Robotics*, 2004.

[34] A. Girard and G. J. Pappas, "Approximation metrics for discrete and continuous systems," *IEEE Transactions on Automatic Control*, Mar. 2005, to appear.

[35] ——, "Hierarchical control using approximate simulation relations," in *Proc. IEEE Conference on Decision and Control*, 2006.

[36] K. Y. Goldberg, "Orienting polygonal parts without sensors," *Algorithmica*, vol. 10, pp. 201–225, 1993.

[37] K. Y. Goldberg and M. T. Mason, "Bayesian grasping," in *Proc. IEEE International Conference on Robotics and Automation*, 1990.

[38] L. J. Guibas, R. Motwani, and P. Raghavan, "The robot localization problem," in *Proc. Workshop on the Algorithmic Foundations of Robotics*, K. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, Eds. Wellesley, MA: A.K. Peters, 1995, pp. 269–282.

[39] J. E. Hopcroft, J. D. Ullman, and R. Motwani, *Introduction to Automata Theory, Languages, and Computation*, 2nd ed. Reading, MA: Addison-Wesley, 2000.

[40] P. Jensfelt and S. Kristensen, "Active global localisation for a mobile robot using multiple hypothesis tracking," *IEEE Transations on Robotics and Automation*, vol. 17, no. 5, pp. 748–760, Oct. 2001.

[41] T. Kameda, M. Yamashita, and I. Suzuki, "On-line polygon search by a six-state boundary 1-searcher," Apr. 2003, submitted for publication.

[42] I. Kamon, E. Rivlin, and E. Rimon, "Range-sensor based navigation in three dimensions," in *Proc. IEEE International Conference on Robotics and Automation*, 1999.

[43] M.-Y. Kao, J. H. Reif, and S. R. Tate, "Searching in an unknown environment: An optimal randomized algorithm for the cow-path problem," in *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 1993, pp. 441–447.

[44] R. M. Karp, "On-line algorithms versus off-line algorithms: How much is it worth to know the future?" in *Proceedings World Computer Congress*, 1992.

[45] J. M. Kleinberg, "The localization problem for mobile robots," in *Proc. IEEE Symposium on Foundations of Computer Science*, 1994, pp. 521–531.

[46] S. Koenig, A. Mudgal, and C. Tovey, "An approximation algorithm for the robot localization problem," in *Proc. ACM-SIAM Symposium on Discrete Algorithms*, 2006.

[47] H. W. Kuhn, "Extensive games and the problem of information," in *Contributions to the Theory of Games*, H. W. Kuhn and A. W. Tucker, Eds. Princeton, NJ: Princeton University Press, 1953, pp. 196–216.

[48] K. N. Kutulakos, C. R. Dyer, and V. J. Lumelsky, "Provable strategies for vision-guided exploration in three dimensions," in *Proc. IEEE International Conference on Robotics and Automation*, 1994, pp. 1365–1371.

[49] A. M. Ladd, K. E. Bekris, A. P. Rudys, D. S. Wallach, and L. E. Kavraki, "On the feasibility of using wireless Ethernet for indoor localization," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 3, pp. 555–559, June 2004.

[50] S. M. LaValle, *Planning Algorithms*. Cambridge, UK: Cambridge University Press, 2006, also available at http://planning.cs.uiuc.edu/.

[51] S. M. LaValle and S. A. Hutchinson, "An objective-based framework for motion planning under sensing and control uncertainties," *International Journal of Robotics Research*, vol. 17, no. 1, pp. 19–42, Jan. 1998.

[52] S. Lenser and M. Veloso, "Sensor resetting localization for poorly modelled mobile robots," in *Proc. IEEE International Conference on Robotics and Automation*, 2000.

[53] T. Lozano-Pérez, M. T. Mason, and R. H. Taylor, "Automatic synthesis of fine-motion strategies for robots," *International Journal of Robotics Research*, vol. 3, no. 1, pp. 3–24, 1984.

[54] V. Lumelsky and S. Tiwari, "An algorithm for maze searching with azimuth input," in *Proc. IEEE International Conference on Robotics and Automation*, 1994, pp. 111–116.

[55] V. J. Lumelsky and A. A. Stepanov, "Path planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape," *Algorithmica*, vol. 2, pp. 403–430, 1987.

[56] T. Mehta, F. Delmotte, and M. Egerstedt, "Motion alphabet augmentation based on past experiences," in *Proc. IEEE Conference on Decision and Control*, 2006.

[57] M. Moll and M. Erdmann, "Manipulation of pose distributions," *International Journal of Robotics Research*, vol. 21, no. 3, pp. 277–292, 2002.

[58] T. Murphey, "Motion planning for kinematically overconstrained vehicles using feedback primitives," in *Proc. IEEE International Conference on Robotics and Automation*, 2006.

[59] C. Ó. Dúnlaing and C. K. Yap, "A retraction method for planning the motion of a disc," *Journal of Algorithms*, vol. 6, pp. 104–111, 1982.

[60] J. M. O'Kane, "Global localization using odometry," in *Proc. IEEE International Conference on Robotics and Automation*, 2006.

[61] J. M. O'Kane and S. M. LaValle, "Almost-sensorless localization," in *Proc. IEEE International Conference on Robotics and Automation*, 2005.

[62] ——, "On comparing the power of mobile robots," in *Robotics: Science and Systems*, 2006.

[63] ——, "Sloppy motors, flaky sensors, and virtual dirt: Comparing imperfect ill-informed robots," 2007, submitted to *IEEE International Conference on Robotics and Automation*. Under review.

[64] ——, "Localization with limited sensing," *Conditionally accepted to IEEE Transactions on Robotics*, Sept. 2006.

[65] G. Owen, *Game Theory*. New York: Academic, 1982.

[66] C. H. Papadimitriou, "Games against nature," *Journal of Computer and System Sciences*, vol. 31, pp. 288–301, 1985.

[67] C. H. Papadimitriou and M. Yannakakis, "Shortest paths without a map," *Theoretical Computer Science*, vol. 84, pp. 127–150, 1991.

[68] M. Rao, G. Dudek, and S. Whitesides, "Randomized algorithms for minimum distance localization," in *Proc. Workshop on the Algorithmic Foundations of Robotics*, 2004, pp. 265–280.

[69] K. Romanik and S. Schuierer, "Optimal robot localization in trees," in *Proc. Symposium on Computational Geometry*, 1996, pp. 264–273.

[70] H. Shatkay and L. P. Kaelbling, "Learning topological maps with weak local odometric information," in *Proc. International Joint Conference on Artificial Intelligence*, 1997, pp. 920–927.

[71] R. Simmons and S. Koenig, "Probabilistic robot navigation in partially observable environments," in *Proc. International Joint Conference on Artificial Intelligence*, 1995, pp. 1080–1087.

[72] M. Sipser, *Introduction to the Theory of Computation*. Boston, MA: PWS, 1997.

[73] D. D. Sleator and R. E. Tarjan, "Amortized efficiency of list update and paging rules," *Communications of the ACM*, vol. 28, pp. 202–208, 1985.

[74] K. Sugihara, "Some location problems for robot navigation using a simple camera," *Comp. Vis., Graphics, & Image Proc.*, vol. 42, no. 1, pp. 112–129, 1988.

[75] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[76] B. Tovar, L. Guilamo, and S. M. LaValle, "Gap Navigation Trees: Minimal representation for visibility-based tasks," in *Proc. Workshop on the Algorithmic Foundations of Robotics*, 2004.

[77] A. F. van der Stappen, R.-P. Berretty, K. Goldberg, and M. H. Overmars, "Geometry and part feeding," in *Sensor Based Intelligent Robots*, 2000, pp. 259–281.

[78] G. Weiss, C. Wetzler, and E. von Puttkamer, "Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1994.

[79] D. Whitney, "Real robots don't need jigs," in *Proc. IEEE International Conference on Robotics and Automation*, 1986.

[80] J. Wiegley, K. Goldberg, M. Peshkin, and M. Brokowski, "A complete algorithm for designing passive fences to orient parts," *Assembly Automation*, vol. 17(2), Aug. 1997.