Constructive Recognizability for Task-Directed Robot Programming*

Bruce Randall Donald
James Jennings
Russell Brown

TR 91-1253
December 1991

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This paper describes research done in the Computer Science Robotics and Vision Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grants No. IRI-8802390, IRI-9000532 and by a Presidential Young Investigator award to Bruce Donald and in part by the Air Force Office of Sponsored Research, the Mathematical Sciences Institute and AT&T Bell Laboratories.

To Appear:
Jour. Robotics and Autonomous Systems
Elsevier/North-Holland.

Constructive Recognizability for Task-Directed Robot Programming

Computer Science Department

Cornell University

Bruce Randall Donald

James Jennings

with contributions from Russell Brown

Preface

I think that everyone who runs a lab should write a "thesis proposal" every couple of years. I enlisted the help of my students to write this one. We write about our progress on mobile autonomous agents. and about our vision of what we should do in the robotics lab over the next few years.

This paper is based largely on the foundational work on which Jim Jennings and I have been collaborating, represented in the papers [DJ, DJ2-4]. It also contains ideas from my work with Russell Brown [BD] and Jonathan Rees [RD]. Other members of the Computer Science Robotics and Vision Laboratory also have helped us with their ideas and by building robots. We thank them all, and we are particularly grateful to Dan Huttenlocher, Pat Xavier, Amy Briggs, Craig Becker, and Mark Lee.

Bruce Randall Donald
Ithaca, 1991

This paper describes research done in the Computer Science Robotics and Vision Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grants No. IRI-8802390, IRI-9000532 and by a Presidential Young Investigator award to Bruce Donald, and in part by the Air Force Office of Sponsored Research, the Mathematical Sciences Institute, and AT&T Bell Laboratories.

Abstract

The primary goal of our research is task-level planning. We approach this goal by utilizing a blend of theory, implementation, and experimentation. We investigate task-level planning for autonomous agents, such as mobile robots, that function in an uncertain environment. These robots typically have very approximate, inaccurate, or minimal models of the environment. For example, although the geometry of its environment is crucial to determining its performance,[1] a mobile robot might only have a partial, or local "map" of the world. Similarly, the expected effects of a robot's actuators critically influence its selection of actions to accomplish a goal, but a robot may have only a very approximate, or local predictive ability with regard to forward-simulation of a control strategy. While mobile robots are typically equipped with sensors in order to gain information about the world, and to compensate for errors in actuation and prediction, these sensors are noisy, and in turn provide inaccurate information. We investigate an approach whereby the robot attempts to acquire the necessary information about the world by planning a series of experiments[2] using the robot's sensors and actuators, and building data-structures based on the robot's observations of these experiments. A key feature of this approach is that the experiments the robot performs should be driven by the information demands of the task. That is, in performing some task, the robot may enter a state in which making progress towards a goal requires more information about the world (or its own state). In this case, the robot should plan experiments which can disambiguate the situation. When this process is driven by the information demands of the task, we believe it is an important algorithmic technique to effect task-directed sensing. This introductory survey article discusses:

1. A theory of *sensor interpretation and task-directed planning using perceptual equivalence classes*, intended to be applicable in highly uncertain or unmodelled environments, such as for a mobile robot.

2. Algorithmic techniques for modeling geometric constraints on recognizability, and the building of internal representations (such as maps) using these constraints.

3. Explicit encoding of the information requirements of a task using a *lattice* (information hierarchy) of *recognizable sets*, which allows the robot to perform experiments to recognize a situation or a landmark.

4. The synthesis of robust mobot programs using the geometric constraints, *constructive recognizability* experiments, and uncertainty models imposed by the task.

We discuss how to extend our theory and the geometric theory of planning to overcome challenges of the autonomous mobile robot domain. One of our most important goals is to show how our theory can be made constructive and algorithmic. We propose a framework for mobot programming based on *constructive recognizability*, and discuss why it should be robust in uncertain environments. Our objective is to demonstrate the following: When

[1] I.e., what the geometry is will have a key role in determining the robot's actions or behavior.

[2] The robot (not the researchers!) performs the experiments, to gain information about the world.

recognizability is thusly constructive, we naturally obtain task-directed sensing strategies, driven by the information demands encoded in the structure of the recognizable sets.

A principled theory of sensing and action is crucial in developing task-level programming for autonomous mobile robots. We propose a framework for such a theory, providing both a precise vocabulary and also appropriate computational machinery for working with issues of information flow in and through a robot system equipped with various types of sensors and operating in a dynamic, unstructured environment. We are implementing the theory and testing it on mobile robots in our laboratory.

Contents

1 Overview

The possibilities engendered by the concept of "robot" are almost limitless. However, unless major scientific advances are achieved to make robots more useful, flexible, and practical these visions will never become realities. If the present course is maintained, robots will continue to be utilized far below their capacity, limited to such tasks as spray painting, spot welding, and palletizing. We view robotics as the science of representing, reasoning about, and manipulating physical objects. We believe that basic research in robotics science can infuse new techniques into mainstream robotics, and greatly enhance the usefulness and practicality of robots.

The primary goal of our research is task-level planning. We approach this goal by utilizing a blend of theory, implementation, and experimentation. We investigate task-level planning for autonomous agents, such as mobile robots, that function in an uncertain environment. These robots typically have very approximate, inaccurate, or minimal models of the environment. For example, although the geometry of its environment is crucial to determining its performance, a mobile robot might only have a partial, or local "map" of the world. Similarly, the expected effects of a robot's actuators critically influence its selection of actions to accomplish a goal, but a robot may have only a very approximate, or local predictive ability with regard to forward-simulation of a control strategy. While mobile robots are typically equipped with sensors in order to gain information about the world, and to compensate for errors in actuation and prediction, these sensors are noisy, and in turn provide inaccurate information. In our previous and current work, we have considered task-level planning in what might best be called *semi-structured* environments. These environments—for example, one typically thinks of a factory, where one might control some of the uncertain factors—are also fraught with modeling, sensing, and actuation uncertainty. However, it is often reasonable to assume a parametric form of uncertainty—for example, that the geometry of the world is known to some tolerance, or up to some parameters, which must in turn be estimated (see, eg., [Don90]). For example, the robot might know the number and position of parts in an assembly, up to some tolerance in configuration, and it might know the shape of each part, up to some tolerances in a parametric design space. In the domain of mobile robotics, this kind assumption is far less valid; we might say there is *structural* uncertainty about the world. One solution is to consider open-loop or sensorless strategies (eg, [EM, EMV]). Another is to attempt to acquire the necessary information about the world by planning a series of experiments using the robot's sensors and actuators, and building data-structures based on the robot's observations of these experiments. This is the approach we investigate. A key feature of this approach is that the experiments the robot performs should be driven by the information demands of the task. That is, in performing some task, the robot may enter a state in which making progress towards a goal requires more information about the world (or its own state). A typical example would involve recognition: *Does this history of sensor and control values indicate the robot been following the correct wall on its map, or has it reached an unanticipated dead-end?* In this case, the robot should plan experiments which can disambiguate the situation. When process is driven by the information demands of the task, we believe it is an important algorithmic technique to effect task-directed sensing.

As a key step towards task-directed sensing, we are developing a theory of perceptual aliasing. This theory is intended to help us cope with the way that distinct features in the world may be perceptually indistinguishable to the robot. The basic idea is that the robot observes the world through its sensors, and information about the world is in effect viewed as through a *projection*. Hence, two features in the world may project to the same sensor readings, and be perceptually indistinguishable (at that time). So, we may say that locally, these two features are *perceptually equivalent*. Landmarks—that is, recognizable places in the environment—arise as a kind of perceptual equivalance class. Now, naturally, by moving, retaining history (state), and filtering or processing information over time, the robot may be able to disambiguate two features which were initially (locally) perceptual equivalent. However, it turns out that, even taking history, motion, and computation into account, one obtains a constructive "calculus of the observable" that characterizes the recognizable landmarks in the world. Our theory of perceptual aliasing is based on this calculus. The basic building blocks of this theory should be the "recognizable sets" (cf. [Can, LMT, Erd])—that is, the places in the world the robot can recognize. One of our most important goals is to show how our theory can be made constructive and algorithmic. This has immediate applications: for example, a map of landmarks is a constructive representation of the history of recognizable sets the robot has encountered. Our goal is to demonstrate the following: When recognizability is thusly constructive, we naturally obtain task-directed sensing strategies, driven by the information demands encoded in the structure of the recognizable sets.

1.1 Methods

The heart of this research is three-pronged. We will use a geometric theory of manipulation and planning where appropriate, to synthesize robot strategies based on approximate models of the world, and explicit modeling of sensor and control error [Erd, Don90]. However, we believe the window of direct applicability for this theory may be rather narrow in the mobile robot domain: we look to this theory primarily for insight, tools, algorithms, and for a mathematical framework in which to analyze our systems and predict their behavior and performance. In particular, there are many facets of the mobile robot domain in which our models of the the world's geometry, or our predictive ability for the robot's actions, are too weak to apply this theory. In this case, the robot's ability to predict its behavior and thereby select actions to make progress towards the goal, is cast into doubt. For this situation believe we can sometimes employ techniques from our theory of Error Detection and Recovery (EDR) [Don90], which is essentially a computational method for synthesizing sensor-based strategies that achieve the goal when it is recognizably reachable, and signal failure otherwise. However, this family of EDR techniques is also dependent on an essentially "parametric" model of error than may be unobtainable for a mobile robot in an uncertain domain. Hence, again, the (direct) applicability of the EDR thory is also circumscribed, although clearly some constructive and algorithmic form of EDR is essential to our approach. One of our goals is to explore practical and theoretical issues in developing Error Detection and Recovery (as a discipline) for autonomous mobile robots in uncertain environments. For all of these reasons, we are developing a theory of sensor interpretation and task-directed planning using perceptual equivalence classes, intended to be applicable in highly uncertain or unmodelled environments, such as for a mobile robot. We combine these techniques into a theory for synthesizing robot strategies and for reasoning about and manipulating physical objects in highly uncertain environments.

The goal of this research is to build autonomous mobile robots and to program them at the task-level. To implement our theories, we are developing techniques that impact the geometric theory of manipulation, task-level planning, task-directed sensing, computational geometry, embedded systems, and a number of technologies for building robust autonomous robot vehicles. We want to develop the underlying theory of task-directed sensing and planning, build up a base of scientific principles for mobile robotics, implement and test the theory in four autonomous mobile robots that we are currently building and programming in our lab, conduct laboratory experiments to evaluate how our theory enhances our ability to program autonomous agents at the task-level, and to measure the robustness of the task-directed sensing and planning systems we engineer (See sec. 9).

2 A Research Agenda

A principled theory of sensing and action is crucial in developing task-level programming for autonomous mobile robots. We propose a framework for such a theory, providing both a precise vocabulary and also appropriate computational machinery for working with issues of information flow in and through a robot system equipped with various types of sensors and operating in a dynamic, unstructured environment.

Structured environments, such as those found around industrial robots, contribute towards simplifying the robot's task because a great amount of information is encoded, often *implicitly*, into the robot's control program. There remains considerable structure, however, in human-occupied environs, including offices and laboratories. Unfortunately, much of the corresponding useful information is lost when the world is projected through imprecise sensors. Furthermore, acquiring information is often mediated by imperfect controls. The key issue, then, is coping with uncertainty.

In recent years, robotics researchers have focused considerable attention on the problem of uncertainty (at many levels): for example, error or noise in modeling, actuation, and sensing [LMT, Mason, DW, Brooks, Mat, Brost, Buc, Caine, Can, Erd, Don, LLS].[3] The domain of applicability for strategies for reducing uncertainty, gaining information, and directing actions to accomplish manipulation tasks is very broad. It includes (for example) (a) Manipulation and Mechanical Assembly, (b) Mobile robot navigation and planning, and (c) Design and Layout. In assembly, we may have sensing and control uncertainty, and sparse or incomplete environmental models. Work on synthesizing assembly strategies is particularly applicable during fine-motion and tight assemblies. Mobile robots may have noisy sensors and operate in dynamic, uncertain environments. In design for assembly, we wish to develop a physical device that will be functional in a variety of physical situations; this "variety" may be viewed as a parametric form of uncertainty.

A central theme to previous work has been to determine what information is required to solve a task, and to direct a robot's actions to acquire that information. In this paper, we expand on this theme, with particular emphasis on strategies that tolerate (and recover

[3]See app. C and also [DJ, DJ2, DJ3] for a detailed discussion of previous work.

from) failure. In addition, one of our goals is to develop our framework such that we can measure the sensitivity of plans to particular assumptions about the world, and in fact to minimize those assumptions where possible.

Briefly stated, our goals include:

- development of a theory of task-level programming of autonomous agents in unstructured environments,
- construction of theoretical tools for answering such information-related questions as, e.g.:
    1. What information is needed by a particular robot to accomplish a particular task?
    2. How may the robot acquire such information?
    3. What properties of the world have a great effect on the fragility of a robot plan/program?
    4. What are the capabilities of a given robot (in a given environment or class of environments)?
- Development of algorithms to implement this theory, and experiments using physical robots to develop, refine, and test the theory.

A key issue, given these goals, is to characterize the kinds of representations and data structures a mobile robot can build in an uncertain and dynamic world—based on its observations of this world though noisy sensors. One key question is how the robot should use *a priori* information about, or "models" of the world, and concomitantly, what are the kinds of errors and mistakes it might make by basing inferences on "matches" between noisy sensor data and (possibly faulty) "expectations" about the world (i.e., models). One way to explore this issue is to investigate how the robot can function with "minimal" models of the world. We often take this route, not so much because we believe this will result in better robots, but rather, because determining what initial state a robot needs to perform a task, and analyzing how the robot should use this *a priori* information are both exceedingly difficult. Indeed, this difficulty drives our investigation. Finally, we may not even have a choice of whether or not to give the robots models of the world: they may be unavailable or highly inaccurate (for example, a mobile robot in an unknown environment), or extremely expensive to use (for example, see [CR, Can]).

3 A Calculus of the Observable

We wish to construct a theory of mobile robotics from a "calculus of the observable." The advantages of such a theory would be that, first, any representations or data structures that the theory requires will *a priori* be only things that it can construct from its observations. We call this property a *constructive theory of recognizability.* Second, if there is a correspondence between (parts of) the robot's data structures, and features or parts of the world, then the resolution of this correspondence will be no finer than the perceptual limit of the robot.

Third, all the "places" in the world that the robot can "address" or "plan for" would be *a priori* recognizable.

We attack the research programme above as a principled exercise in spatial reasoning and experimentation. For example, one of our first tasks is to suggest precise definitions of the terms "resolution" and "perceptual limit" used in the paragraph above. We believe the benefits of our approach should be a framework for mobile robotic planning and reasoning which features *task-directed sensing*: that is, the sensing and actions that the robot undertakes are driven by the quantity and quality of its needs for information about the world. This paper concentrates on the basic concepts and constructions in our theory. In [DJ,DJ2-3] we also discuss algorithms for computing these representations, both incrementally (from the world), and off-line (given a model).

4 An Architecture for Constructing Recognizability

The methods we propose may be described in terms of three layers, each building on the next. The first, definitional, layer consists mainly of the consequences of a principled, functional approach to examining robotic sensing. We begin with a functional definition of the action of a sensor. That is, a sensor is a projection from the world (and the robot's configuration) onto a sensor space. Issues of perceptual equivalence, perceptual aliasing, recognizability, sensor uncertainty (noise), and robot system capabilities arise here. An important feature of this approach is its close connection with a theory of Error Detection and Recovery (EDR) [Don, Don90]. In fact, robot programs developed within our framework exhibit "natural" EDR properties. For example, the goal (and subgoal) states of a mobile robot control program, when developed within our framework, are *a priori* recognizable. Thus, error detection is built-in.

When sensors are modeled functionally, one might think of ways to make use of various compositions of sensing functions. In fact, one may define "virtual sensors" in terms of existing "concrete sensors". We will describe a method for task-directed construction of such "virtual sensors". Virtual sensors are queried in robot programs much as their concrete counterparts are. In allowing the task to direct the composition of virtual sensors, we derive robot programs which are organized in such a way as to guide the robot toward acquiring the information it needs to accomplish the task; i.e. many information-acquisition and representational issues are made explicit. For example, consider a manuipulation task: the task of pushing a box stably along a wall. The mobot needs a way to sense the relative position of the wall and box, and also the dynamic relationship of the mobot to the box. Now, the motivation for constructing a "wall sensor" is simply that the task might have been described (i.e., specified) in human-oriented terms such as walls, boxes, etc. These perceptual categories may be indistinguishable to the robot with noisy sensors. In actuality, the robot will sense features of the world that are perceptually equivalent to some human-programmed notion of a wall. We can provide the robot with a virtual wall sensor, that is, a sensing and computation tree that combines current sensory information, history (retained state), and computation, and outputs a parameterized "inference" about recognizing an object in the same perceptual category as "wall." This "inference," then has two parts: a *symbolic* part, that identifies the inference with the task-level specification of a "wall," and a *parametric* part, that represents the parameters that can vary with respect to how the robot perceives a wall—eg, relative distance, orientation, extent, etc. Now we ask, how might this wall sensor work, and how would it be used in constructing a virtual sensor appropriate to this task?

First, a time-indexed history of the robot's controls may be integrated into an odometry sensor, and when odometry is in turn combined with the output of various range sensors, the robot may sense "$\texttt{wall}(\alpha_1, \ldots, \alpha_n)$." For example, suppose the robot senses a range contact dead ahead. It can verify the hypothesis that this range contact represents a (piece of a) wall by attempting to move parallel to the hypothesized wall, and observing whether the range sensor returns the same (or nearly the same) orientation and distance readings. This is the sensing strategy that the virtual sensor for $\texttt{wall}(\alpha_1, \ldots, \alpha_n)$ encodes, and the output of the sensor would be (in the case of success), the hypothesis of the existence of a wall, and an estimate as to its parameters. The key point is that this process of sensing, computation, retention of state, and the recursive construction and conditioning of virtual and concrete sensors can be viewed as a general geometric exercise in task-directed sensing. Our mission here is to elaborate that mechanism.

Hence, in particular, the intended "meaning" of $\texttt{wall}(\alpha_1, \ldots, \alpha_n)$ is that the robot is near a wall with parameters $(\alpha_1, \ldots, \alpha_n)$, which describe the location, extent, and possibly other parameters of a wall. We have force and torque sensors attached to a "manipulation bumper" (for pushing) on the mobot. Next, the output these force sensors is combined with contact sensor information to report on the experimental dynamics of the robot's interaction with the environment. Suppose that a virtual box sensor has been defined in a similar manner to the wall sensor described above. A contact sensor can then detect contact with the box, and force and torque sensors report on the robot's interaction with the box. While translating along the wall in a stable way, the box stays in the same relative position to the robot. In addition, some nominal force and torque is required to continue moving the box; a large required force may mean that the box is stuck. The sensing strategy encoded by a box-pushing virtual sensor would output the relative location of the box, and also the dynamical properties associated with pushing it, such as point of contact, and applied force and torque. This sensor may also output the relative "drift" of the center of friction of the box relative to the robot. Such information could be used to alter the robot's controls such that it would continue to push the box without rotating it.

Finally, we may construct a task-directed virtual sensor that incorporates the wall sensor and the experimental dynamics of the robot's interaction with the box. This sensor is built such that it reports that the robot has achieved a goal state, has entered a terminal non-goal (failure) state, or that the robot should continue. Note the intrinsic EDR character of the resulting "program," in which goal and failure states are recognized when they are reached.

Information from physical sensors, previous state, and computation results, combine as task-directed virtual sensors are constructed. The flow of information through the hierarchy of such virtual sensors is quite explicit. Methods of construction of virtual sensors comprises the second layer of our approach.

The third layer describes how the tools we have built, such as virtual sensors, may be used to do task-level programming. The emphasis here is on how to make use of multiple internal representations of the world. We propose to support different representations using a *version-space*-like technique [Mit] to maintain data structures with respect to current and past observations.

5 On Perceptual Equivalence

We introduced our ideas in a research programme outlined in [DJ,DJ2-3]. We present in this section some of the main ideas. The presentation here is simplified to emphasize the main points. Key additional components include (a) how to model more complicated sensors and (b) the impact of sensor noise. Some of the theory behind our treatment of control and sensing uncertainty can be found in [Don90, Erd, E89]; for the interested reader, we have provided examples illustrating these issues and our approach in appendices A and B. Here is the idea behind how we model sensors:

Definition 5.1 (DJ) *For a world M, a point x in configuration space C, and a sensor space B, the* world sensing map *p_x is given by $p_x : M \to B$. It may be globalized into a* world-sensing bundle *$p : C \times M \to B$ by the rule $p(x, y) = p_x(y)$.*

The mapping p_x induces a partition of the world into equivalence classes of points that will appear the same under the sensor. For example, consider a range sensor (without noise).[4] We can represent the sensor "value" b for $y \in M$ as being one of the following:

1. An integer $\lfloor d \rfloor$, if there is an obstacle at y and there are no obstacles on the line segment $[x, y)$.
2. The value $\emptyset$ if there are no obstacles on the line segment $[x, y]$.
3. The value ∞ if there is another obstacle on the line segment $[x, y)$.

So, if Z^+ denotes the non-negative integers, we can view the space B of sensor values as $B = Z^+ \cup \{\emptyset, \infty\}$.

Sensors are modeled, then, as functions which project the world, M, (and the robot's configuration, C) onto a sensor space, B. Note that for this example, we do not choose to model the range finder as a simple mapping from controls to (say) integers representing, eg., time-of-flight for a sonar ping. That is because we wish to define a sensing map that encodes the *information* obtained in the return ping. The information contained in a (non-infinite) return ping is not simply "There exists an obstacle at distance d." It is considerably more:

> *(*) "There exist no obstacles up to distance d; there exists an obstacle at distance d; and beyond d we have no information."*

Hence, we developed this sensor map as follows. We began by asking the question:

> *(Q1) Given a point y in the world M, what effect does y have on the values we sense?*

We call the answer to this question the *pointwise* definition of the sensing map: it gives us the rule $p_x(y) = b$. In app. A we present more details of how more complex physical sensors may be modeled in this way.[5]

[4] The simplified framework we describe here can be augmented to handle noise. See appendix B.

[5] We also note that the value b could represent a probability distribution, as in [ME, Mor]. See appendix A.

5.1 Perceptual Limits

Now, a natural question to ask is: for a given sensing map p_x, what are the finest possible perceptual equivalence classes? In this case, the answer is clear: the finest possible perceptual equivalence classes the robot could distinguish are the unit annuli centered on x with integral inner and outer radii. See fig. 3. Now, call this partition (into unit annuli) $\overline{\mathcal{U}}$. We notice that $\overline{\mathcal{U}}$ is at *least as fine* as any partition of the world into perceptual equivalence classes that we will encounter given some *particular* geometry. For example, $\overline{\mathcal{U}}$ is finer than the partition induced by the geometry in fig 3 and which is induced by sweeping the line in fig. 4 about x into annuli. Here is how one may formalize this notion. Let G_α represent a world geometry, such as the walls and so forth in fig. 3. Now let $\mathcal{U}_\alpha$ denote the resulting partition of the world into perceptual equivalence classes, for example, the annuli swept out by the linear arrangement in fig. 4. For another geometry G_β we obtain a different partition $\mathcal{U}_\beta$ and so on. For any two partitions $\mathcal{U}_\alpha$ and $\mathcal{U}_\beta$, one (say, $\mathcal{U}_\beta$) may be finer than the other ($\mathcal{U}_\alpha$), or else they will be incomparable. When $\mathcal{U}_\beta$ is finer than $\mathcal{U}_\alpha$, we write $\mathcal{U}_\alpha \hookrightarrow \mathcal{U}_\beta$. To say that $\overline{\mathcal{U}}$ is the *finest* partition possible for p_x is to say that for any geometry $G_\alpha, \mathcal{U}_\alpha \hookrightarrow \overline{\mathcal{U}}$. Of course, there always exists some such $\overline{\mathcal{U}}$ trivially (eg, consider the 1-point sets) but the interesting case is where the sets have finite extent (as in this example with p_x). Let G_α range over the set of all geometries. The "finer" relation structures the resulting perceptual equivalence class partitions into a lattice. This lattice is dominated by its direct limit, which is $\overline{\mathcal{U}}$. The map we are considering, p_x, is interesting in that despite the existence of an uncountable infinity of possible world geometries G_α, only a finite number of partitions into perceptual equivalence classes $\mathcal{U}_\alpha$ are possible.[6] The lattice in this case simply reduces to a *finite* directed graph whose nodes are the $\{\mathcal{U}_\alpha\}$ and whose edges are the $\hookrightarrow$ relationships. The graph has one "sink" which is the "minimal" partition $\overline{\mathcal{U}}$ into unit annuli. This direct limit $\overline{\mathcal{U}}$ precisely quantifies the capabilities of the sensing map p_x, and hence we call the partition $\overline{\mathcal{U}}$ the *perceptual limit of the sensing map* p_x.

The perceptual limit has the following constructive significance. For a finite lattice, any obtainable partition $\mathcal{U}_\alpha$ of the world into perceptual equivalence classes can be constructed by finite unions of the classes in the perceptual limit $\overline{\mathcal{U}}$. Hence, we think of a geometry G_α as "controlling" how the classes in $\overline{\mathcal{U}}$ are unioned to obtain $\mathcal{U}_\alpha$.

5.2 Robot Capabilities

In sec. 5.1, we sketched how, once we fixed a sensing map (like p_x), different environments can generate different partitions into perceptual equivalence classes. We suggested that all these partitions can be partially ordered on fineness. The resulting structure is a *lattice*, and it is dominated by its direct limit, which we termed the *perceptual limit* of p_x. The perceptual limit represents the "smallest features" the robot can interdifferentiate, in the best case. So, the perceptual limit is obtained by fixing the sensing map p_x and varying the environment.[7]

We now consider fixing the environment (world geometry), and varying the sensing map. That is, we consider varying: (a) the sensing modalities, (b) the computation used to inter-

[6]The finiteness assumption assumes the range sensor has a maximum range.

[7]In our discussion, this is tantamount to varying the world geometry. However, more generally, one could vary other properties–eg, color, motion, or texture.

pret the sensors, and (c) the amount of "state" or history the computation in (b) can retain. It is important to realize that history (b) and computational power (c) are distinct concepts. *Computational power* (b) is characterized by the time and space complexity of the computation that the robot can perform. *History* (c) is characterized by storage requirements or information complexity. (Compare [E89]). To see this distinction, imagine two extremes:

1. A robot with the power to perform any polynomial-time computation, but which remembers no state.

2. A robot which can remember lots of state but can only perform $O(1)$ work using that state.

Hence robot 1 can be booted or have its storage erased each time-step, without impeding its performance.

We call a sensing map p_x that uses no history a *raw* sensing map. Now suppose that we choose to use history in addition to interrogating the world through p_x. We will see this is equivalent to using a *finer* raw sensing map p'_x (i.e., one that results in a finer partition of the world). That is, loosely speaking, we have that

$$p_x + \text{ history } \cong p'_x.$$

We call the finer map p'_x the *effective* sensing map, hoping that this term resonates with Erdmann's "effective set of sensor interpretations" [E], which he obtained by combining history with sensing. Each change in sensing, computational power, and state can result in a new (effective) sensing map, and hence a new partition of (a particular) world into perceptual equivalence classes. As before, in sec. 5.1, all these partitions (now for a fixed geometry!) can be arrayed in a lattice, and partially ordered on fineness. In [DJ2,BD,RD] we discuss these ideas in some detail and develop applications, an implementation, and experiments. The model is augmented by considering the effect of sensor noise and error. We identify global landmarks, which are the recognizable sets in configuration space, called the C-perceptual equivalence classes.[8] We can compute them, given a world model, and we can give efficient algorithms for doing so. In the absence of a model, these "landmarks" can be acquired incrementally, through exploration. A graph of these explorations, called the *RR-graph,* encodes the reachability relationships between the recognizable sets. Again, it may be computed from a model, but in the absence of a model we can "encode" it using the sensor values and history alone. For a sensor space B, we call this a "B-encoding" of an RR-graph. This representation seems useful for mobots with minimal models.

RR-graphs and C-perceptual equivalence classes are simple concepts; these recognizable sets form the basic building blocks for our theory. These landmarks, or recognizable sets—that is, the places in the world that the robot can recognize and distinguish between—have structure we can exploit in developing sensor-based planning algorithms. In particular, the interaction of the lattice and history is subtle and can be used to plan or program sophisticated strategies for achieving goals and accomplishing tasks.

[8]From definition 5.1, they are the cells in the arangement generated by sets of the form $\pi_C p^{-1}(b)$ for $b \in B$, where π_C is the natural projection of $C \times M$ onto C.

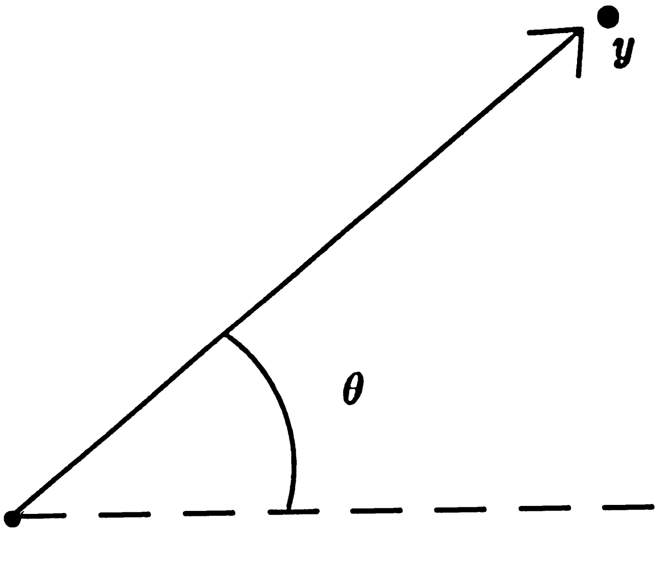

Figure 1: A mobile robot at configuration x is pointing its range finder in orientation θ to interrogate a point y in the environment.

6 Building Virtual Sensors

Consider the general problem of how new sensing maps can be built out of old ones. We can show that the key constructive tools are integration (of sensor histories) and union (of equivalence classes), composed with procedural abstractions similar to `map` and `Accumulate` (see app. A). The motivating reasons for building new sensing maps out of old ones are as follows:

1. It shows how the robot can use information we are confident it can obtain (eg, raw sensor data), to make sound inferences about the existence and geometry of objects in the world that are relevant to its task.

2. If the construction is algorithmic, then we can implement sophisticated sensors using simple ones plus some computation, and be assured that the construction is principled and has some predictive power.

In addition, there is hope we can define quite general "virtual sensors." The idea is as follows. Suppose we wish to specify a robot task in terms of landmarks (perceptual equivalence classes) recognizable to humans. Examples include: the hallways, doors, corners, the atrium, and Hopcroft's office. How can we know if these features in the environment are recognizable to the robot? The key issue here is that if a task is specified in terms of "landmarks" that are not recognizable sets to the robot, then the robot may not be able to achieve the goal—in this case we say the task specification is not *robust*. This notion of robustness is analogous to the stronger notion of a *guaranteed strategy* in the [LMT] framework. Thus we ask, "How can a task be robustly specified to a robot when the robot's perceptual equivalence classes and the human's perceptual equivalence classes are different?"

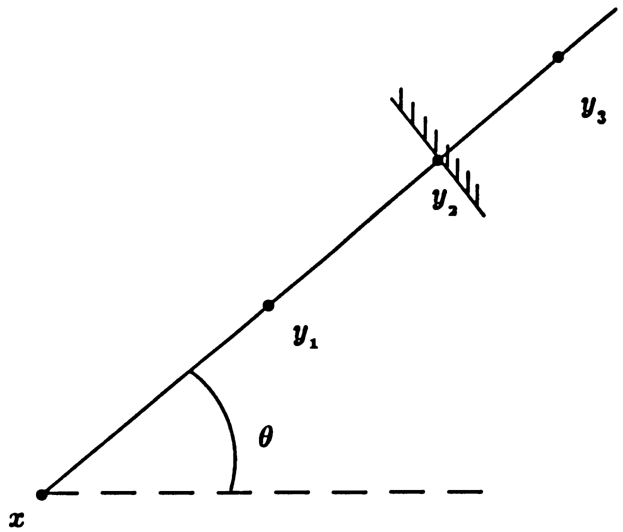

Figure 2: Suppose we have the world $M = \Re^2$. Consider points y_i in the world, which are in direction θ from x. We imagine the robot "interrogating" y_i by pointing the sensor in direction θ. In this example, there is an obstacle at y_2, which is distance d from x. We have $p_x(y_1) = \emptyset$, $p_x(y_2) = \lfloor d \rfloor$, $p_x(y_3) = \infty$.

Next, given the robot's "concrete" on-board sensors (eg, sonar, vision, etc), there will exist landmarks (perceptual equivalence classes) that are recognizable to the robot. But what are they, and how can humans understand them? One possibility is to let the robot rove around and then "dump" its map of perceptual equivalence classes in some human-readable form. Then, the human programmer could correlate (match) these landmarks with the features in the environment with respect to which he can specify the task. This may be a difficult matching problem; we note that tools from model-based recognition in machine vision might be used here. In this vein, we pause to note that, throughout our discussion, the term "human programmer" could be replaced by "task-level planner"—in effect, an automatic programmer. We will say more about this later.

Another approach to robust task specification is as follows. The human defines "virtual sensors"—effectively "oracles" that can recognize the features in the environment (walls, etc) that are critical to task specification. Thus, these virtual sensors are defined to return "symbolic" information like *"There is a wall of length l at coordinates (θ, d)."* The programmer then specifies the task in terms of these recognizable places. Next, the programmer tells the robot how to build or approximate these "virtual sensors" using a combination of raw sensors, computation, and retained state. This is the process we term "building new sensing maps out of old ones." Corresponding to the available effective sensing maps are what we will call the *concrete* perceptual equivalence classes. Our job is to show how to build the *virtual* perceptual equivalence classes corresponding to a virtual sensor out of the concrete perceptual equivalence classes on hand.

We believe that this process is quite general: namely that a new sensing map can be constructed by integration and union (or `map` and `Accumulate`) of existing sensor maps. Roughly speaking, the process involves first passing to the perceptual equivalence classes for the existing sensor maps. Then these classes are unioned, and the union is "controlled"

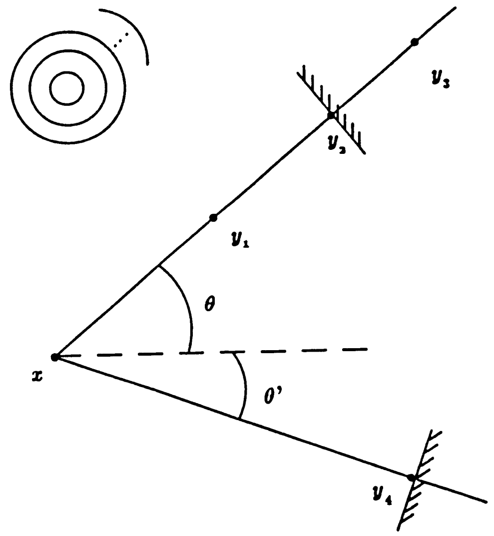

Figure 3: If we disregard θ, two obstacles at roughly distance d will appear the same, even if they are at different angular locations. Hence, since $p_x(y_2) = p_x(y_4)$, y_2 and y_4 are in the same perceptual equivalence class. The *perceptual limit* of this sensing map is given by the concentric annular sets shown here.

by the values of some computation or other sensing maps. This corresponds to constrained integration (boundary or contour integration) of the sensing "signals." As an example, see fig. 8 and sec. A.2 below. This line of inquiry is quite general: it concerns how we can combine differential information relationships using computation to build structures with more information. That is, our research asks what are the differential relationships our sensors measure in the world, and how, over time, can we aggregate them to gain information? We hope that our answer, while still partial, can shed some light on this key robotics problem.

7 Task-Directed Sensing and Planning

When virtual sensors can be defined constructively using concrete sensors, computation, and retained state, we enable a robot using these sensors to perform *task-directed* sensing. That is, if a task is specified in terms of the virtual sensors—more precisely, if the task is specified in terms of the "landmarks" recognizable using these sensors—then the robot may perform experiments to determine which "landmarks" it is near. This is because (i) the landmarks recognizable with the virtual sensors are virtual perceptual equivalence classes, (ii) the virtual perceptual equivalence classes are defined (for the robot) by a data structure that specifies how the concrete sensors should change, with respect to controls, over time. The notion of change here is captured in our framework using the *recognizability gradient.*

We must assume that the "lowest level" concrete perceptual equivalence classes may be reconstructed from sensor values using, for example, built-in sections of the associated sensing map (call this map p). That is, the robot must have some means of hypothesizing the existence and geometry of a perceptual equivalence class such as $\pi_C p^{-1}(b_i)$ given only a

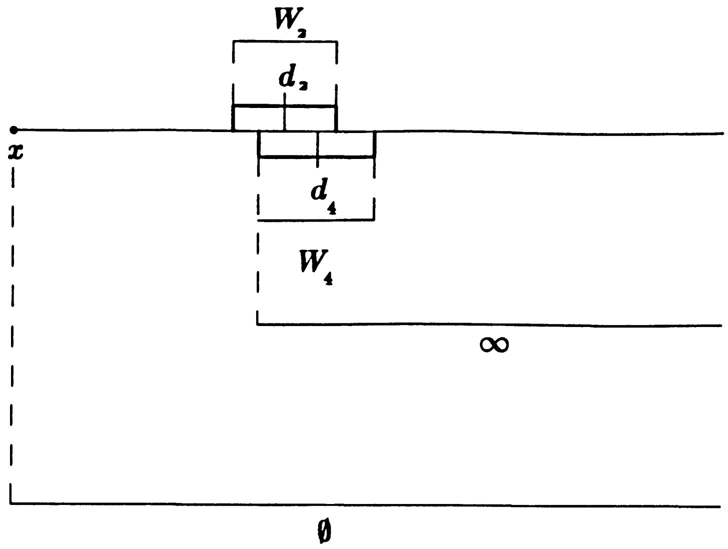

Figure 4: This line represents distances of points $y \in M$ in the world from x. It is annotated with the different sensor values possible for such points under the sensing map p_x. The intervals W_2 and W_4 denote the distances of points in the two walls depicted in fig. 3.

---

history of sensor values $\{ b_i \}$. This capability may be encoded in the form of local sections[9] of the map p. But, once the robot can reconstruct *some* concrete perceptual equivalence classes, then it can begin composing these classes to form virtual classes. The definition of a virtual perceptual equivalence class (say, S, from section A), suggests, via the recognizability gradient, experiments (control changes and corresponding expected sensor readings) to identify S.

But once a virtual sensor p^* and its perceptual equivalence class are defined, we may then repeat the process, and in turn treat p^* as a *concrete* perceptual map. Using this concrete sensing map we may go on to construct another *virtual* sensing map $\wp$ and its perceptual equivalence classes, and so forth. The terms *concrete* and *virtual* are relative, and in fact, all sensing maps are partially ordered on constructibility. This order defines a hierarchy of virtual perceptual equivalence classes; from this point of view it seems clear that the job of an (offline) task-level planner is to make up a hierarchy of virtual perceptual equivalence classes. We contrast this with the job of an (online) "motion planner" whose job is

1. Using local sections, construct concrete perceptual equivalence classes.

2. Perform experiments to match the concrete perceptual equivalence classes against the virtual perceptual equivalence classes.

[9]Given a bundle $p : E \longrightarrow B$ we call a map $s : B \longrightarrow E$ a *section* of p if $p \circ s$ is the identity on B. A section is called "local" if it is only defined on a neighborhood U of B. Suppose we have a partial or incomplete model of the geometry of the world, and a partial or incomplete characterization of the sensor map p. In our view, a "small number" of remembered sensor values and control values defines a kind of local section of the world sensing bundle $p : C \times M \longrightarrow B$. It is possible that a "family" of these sections $\{ s_t \}$, parameterized by time, may be used to construct an approximation to the inverse map p^{-1}.

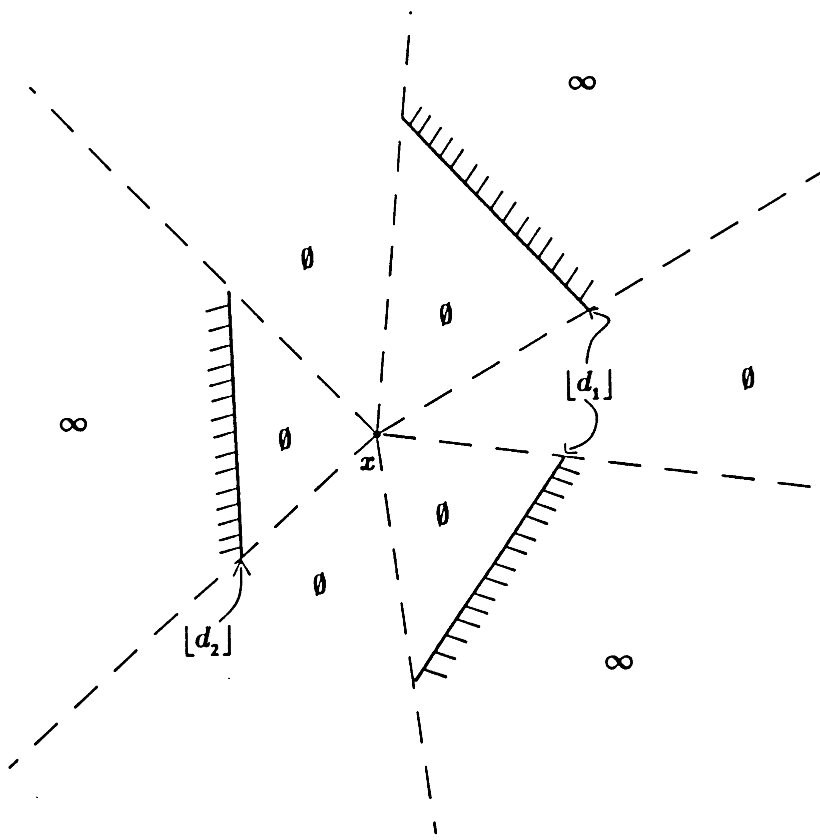

Figure 5: Suppose the robot uses θ, the direction in which the sensor is pointing, to distinguish between obstacles at the same distance, but at a different relative orientation. Then the world (around x) is divided into a finer partition than in fig. 3. Here is an example of a world with three walls. The resulting perceptual equivalence classes are 3 disconnected sets with value ∞, two disconnected linear sets with value $\lfloor d_1 \rfloor$, another linear set with value $\lfloor d_2 \rfloor$, and a large star-shaped non-convex region with value $\emptyset$. (In this example the resolution of angular θ sensing is assumed to be quite coarse).

3. Plan (concrete) perceptual equivalence class-wise controls to accomplish (attain) the virtual perceptual equivalence classes required by the task.

8 Observations and Expectations

We now consider some key problems we must overcome. Our discussion here is considerably abreviated. Suppose we wish to design and program a mobile robot using our framework.[10] Then, we need to determine:

1. What initial state should the robot have?
2. How can the robot support multiple representations (eg, of perceptual equivalence classes from different sensors) of the same object?
3. What "program" is running on the robot? Is this just 1, above? (That is, is the initial state of the robot just the contents of its memory, both data and executable?)

[10]This section owes much to discussions with Tomas Lozano-Pérez, Matt Mason, and Mike Erdmann, and we are grateful to them for their insight and suggestions.

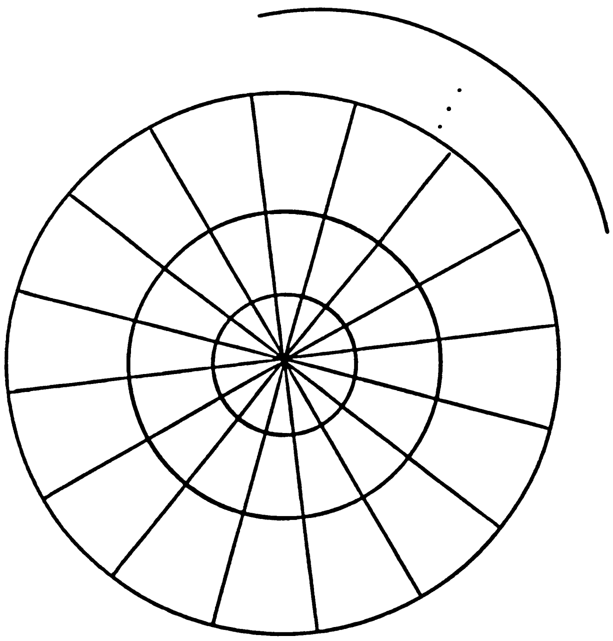

Figure 6: Consider a sensing map p_x^* that takes θ into account. The *perceptual limit* $\overline{\mathcal{U}}^*$ of p_x^* is shown. The perceptual limit is the partition into the finest possible perceptual equivalence classes under this sensing map. Each equivalence class lies on a line through x, has unit length, and starts and ends at an integral distance from x.

4. How can we encode "expectations" about the world, in the form of 1 and 3?

These questions highlight the difference between *expectations* about the world and *observations* of the world. *Observations* are constructed solely from sensor and control history. *Expectations* are a mechanism for using initial state (1 and 3) plus history to match pre-existing data structures against the world, and constructing more state that will in turn be matched against future observations. Grasping the distinction between expectations and observations is crucial to understanding our framework. Compare the work of Cox, Durrant-Whyte, and Leonard [CL,LDWC], who make a similar distinction.

Let us illustrate why resolving these questions is somewhat tricky. Suppose we wish to write a robot *program* that uses or addresses entities in the world such as walls, doors, junctions, corridors, etc. If we pre-encode these features into the initial state (1 and 3, above), then the mobot will attempt to "match" the environment against these templates, and will hallucinate these features even when they are inappropriate. For example, an office robot in the open desert might still build up a "map" with walls, corridors, etc.

First of all, this might not be all bad—for example, an arroyo is a bit like a corridor, and a cliff like a wall. But let us address the undesirable characteristics of matching the world's projection through our sensors, against pre-encoded expectations. How, then, shall we reconcile the two fundamentally different data structures:

A. *"Maps", constructed by a program, that consist of "expected" features matched against observations from the world.*

VERSUS

B. *RR-graphs of perceptual equivalence classes, and Lattices of RR-graphs.*

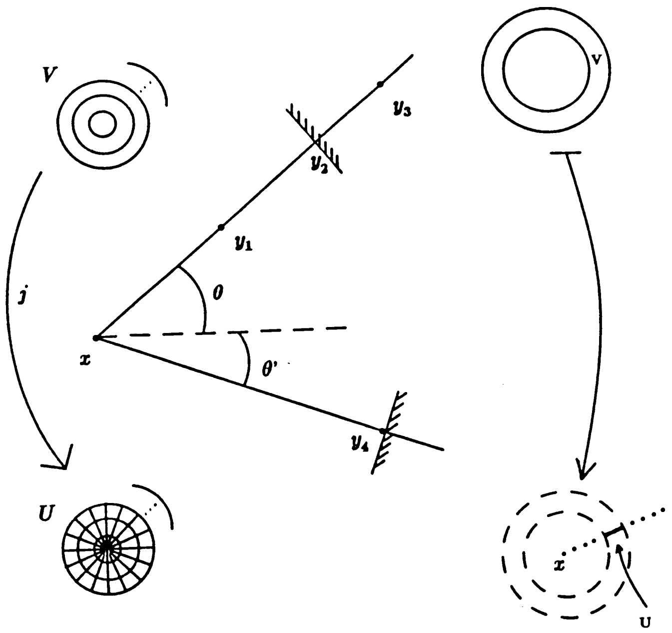

Figure 7: **Illustration of the inclusion map $\mathcal{V} \hookrightarrow \mathcal{U}$. $\mathcal{V}$ is the partition obtained by a sensing strategy that merely interrogates the distance of objects. $\jmath$ models a "better" sensing strategy that interrogates the relative orientation θ as well. The image of equivalence class $V \in \mathcal{V}$ will be a set $U \in \mathcal{U}$ of the form $U = V \cap r(x, \theta)$, where $r(x, \theta)$ is the ray from x in direction θ.**

Clearly, both are useful: for example, (A) has addressable entities such as walls, corridors, etc, that can be used a "primitives" in pre-existing robot programs. However, these "features" crucially depend on the assumptions, encoded in the form of 1 and 3, above. This dependence makes the entities (eg landmarks) in (A) fragile. On the other hand, how can a human programmer know what the entities (the recognizable sets) in (B) look like? For example, we'd like to be able to tell the mobot, *Go down the hall and to the left* without first "microcoding" this high-level instruction in terms of concrete perceptual equivalence classes.

8.1 Multiple Representations

When asked how to reconcile (A) and (B), our answer, in a sense is, "why compromise?" We propose to support both representations. The idea is to keep both representations around and to use a version-space like technique [Mit] to maintain the data structures with respect to current and past observations. Structures like (A) would be "higher" in the lattice, since they are more "optimistic" —they rely on the accuracy of the pre-encoded expectations, and the accuracy of the matching algorithms. If correct, however, they also contain more information, since they encode not only sensor and control history, but also a correlation of that history with landmarks in the environment. On the other hand, structures like (B) are "safe" or "conservative"—they are based only on the fundamental laws of the calculus of the observable. While conservative, they contain less information since they do not match the history to the world. In between (A) and (B), we can imagine various other possible worlds, which form "partial" matches of RR-graphs to expectations. Indeed, one can imagine a partial order of such interpretations. The order is induced by the size of the domain of the matching function, and a measure of its monomorphism (i.e., how "one to one" it is).

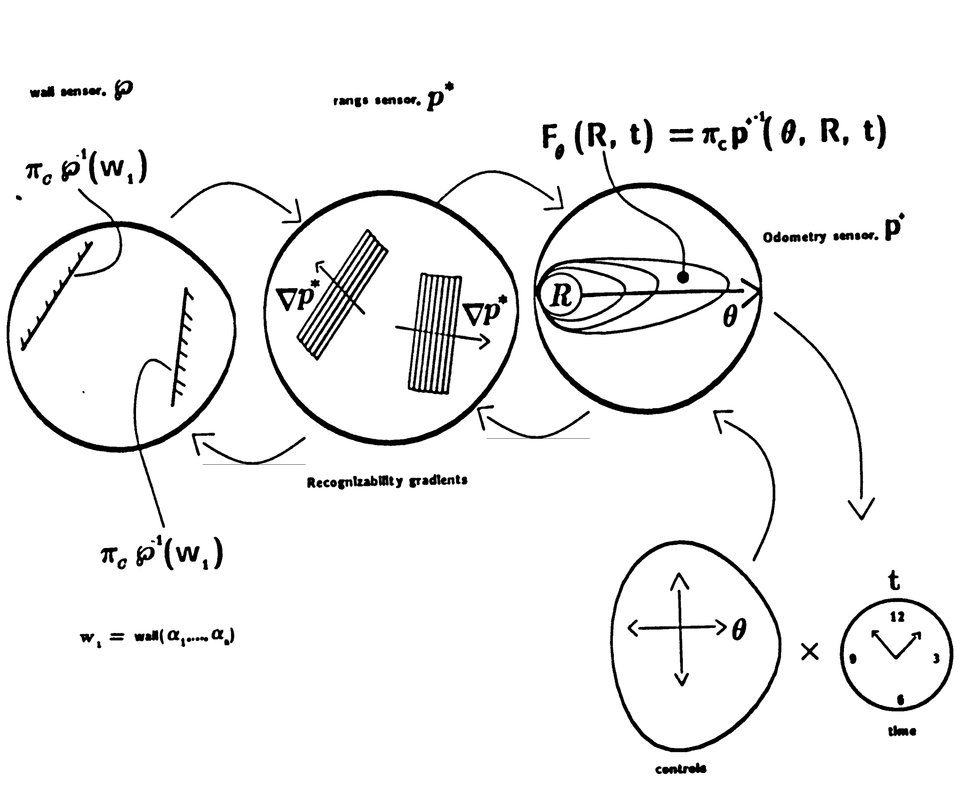

Figure 8: An illustration of how controls and time are integrated to build an odometry sensor $p^\diamond$. Odometry classes may be further integrated with range sensor data, and unioned to form a sensing map p^*, whose perceptual equivalence classes are strips parallel to walls. One may follow the recognizability gradient of these classes, and even define a virtual wall sensor $\wp$. Note that the odometry perceptual equivalence classes correspond to forward projections of the previous classes under the given robot controls and control uncertainties. See app. A and A.2.

Each entry in the lattice is a partition of the world, together with connectivity information. In general, the entries in the lattice correspond to possible worlds, and they are ordered from most optimistic (most information) to least. That is, instead of maintaining just one interpretation, we propose to keep a set, or *range* of interpretations, from safe to optimistic. This set is represented by an intermediate range—(a connected subgraph) in the lattice—of partitions. The top of the lattice is a complete, global world model. The connectivity information at the top corresponds to the ability to perform predictions (eg, physical simulation), with perfect accuracy. The bottom represents knowing nothing about the world. In between is a blob of feasible interpretations. As new observations are made, the mobot prunes interpretations that are inconsistent. For example, our feasible set of interpretations might contain *both* an RR-graph based solely on the history and concrete perceptual equivalence classes experienced by the robot, and *also* a "match" (or several possible matches) of this history to expected landmarks like walls, doorways, and corridors. We wish to build a system that functions as follows: In the office environment, over time, the evidence for the second interpretation (walls, etc.) would increase. In the desert, however, the weight of observation would eventually cause this interpretation to appear more unlikely, and eventually it would be pruned altogether.

To develop such a system, we must be able to show how to encode expectations about the world as perceptual equivalence class-like structures within our framework. We propose that *virtual sensors* provide such an encoding.

8.2 Encoding Expectations

To encode expectations, we define a set of *virtual perceptual equivalence classes.* When matched against observations, these encode "symbolic" information about the world, for example, *"There is a wall of length l at coordinates (θ, d)."* The precise definition is somewhat complex, but we can imagine that it consists of two parts: a "symbolic" predicate like `wall`, and a set of parameters, for example, (θ, l, d) in this case. Examples of predicates for an office robot might reference features in the world such as walls, corridors, or junctures; we may write them as, for example, `wall`, `corridor`, `juncture`, etc. The parameters for these predicates would specify the size, orientation, and so forth, of the feature in the world. Hence, we can imagine that the robot matches its observations against a set of templates (`wall`, etc) and instantiates the parameters of the template in the match. To make this notion precise we also need an *interpretation function.* In logic frameworks for AI, the interpretation function maps these predicates to possible worlds containing the features. Compare, for example, [R]. One may view our framework as specifying a geometric interpretation function. When discussing a general predicate we will imagine it has the general form `wall`(θ, l, d). We'll write a typical predicate like this: `wall`$(\alpha_1, \ldots, \alpha_n)$. So `wall` is the predicate and $(\alpha_1, \ldots, \alpha_n)$ are the parameters. We say that `wall`$(\alpha_1, \ldots, \alpha_n)$ *defines* a (parameterized) virtual perceptual equivalence class; the precise relationship will become clear below.

Our thesis is that the parameterized virtual perceptual equivalence classes encode local sections of the world sensing bundle $p : C \times M \rightarrow B$. Consider the following constructions:

History. We have *sensor history*, which may be represented by a map b^* from time T to sensed values B; hence $b^* : T \rightarrow B$. We have control history, which we might represent (for example) as a map v^* from time to commanded controls (eg, commanded velocity): $v^* : T \rightarrow TC$.

Local Sections. Given a bundle $p : E \rightarrow B$ we call a map $s : B \rightarrow E$ a *section* of p if $p \circ s$ is the identity on B. A section $s_t : U_t \rightarrow C \times M$ is called "local" if it is only defined on a neighborhood U_t of B. If there is a local section s_t for each time t in some interval of T, then we have a parametric family of local sections.

Expectations and Inference. History can define a family of local sections. The map s_t encodes our hypotheses or expectations about what features in the world correspond to the "current" sensor values.

Virtual Perceptual Equivalence Classes. The virtual perceptual equivalence classes encode local sections of p. See fig. 9.

To see the last point, consider the predicate `wall`$(\alpha_1, \ldots, \alpha_n)$ (here, we don't care about the semantics of the parameters $(\alpha_1, \ldots, \alpha_n)$). Here is how we define the virtual perceptual equivalence classes associated with `wall`. For discussion, we fix a set of parameters

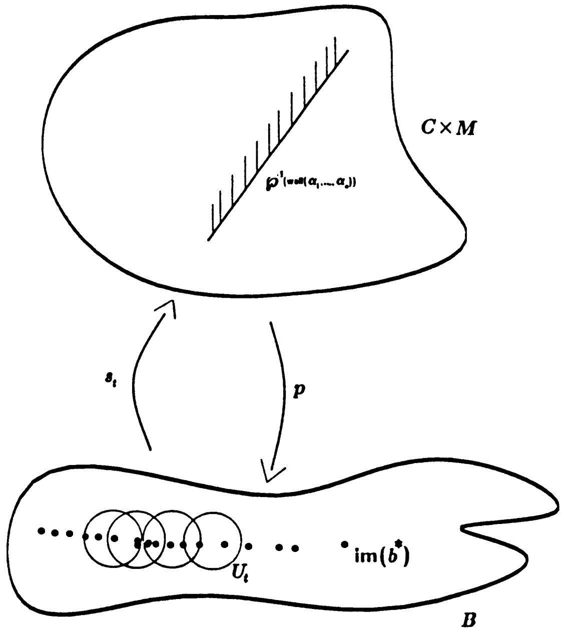

Figure 9: **Shows how the perceptual equivalence class for $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$ is encoded using local sections of the world sensing bundle p, which is a concrete sensor.**

$(\alpha_1, \ldots, \alpha_n)$ for $\mathtt{wall}$. Now, the information encoded in the predicate $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$ is, essentially, *"There is a wall with parameters $(\alpha_1, \ldots, \alpha_n)$."* Now, we imagine the robot as comprising some collection of communicating subsystems. If one subsystem passes another a "message" saying $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$, we can imagine that this message encodes either the "absolute" information *"There is a wall with parameters $(\alpha_1, \ldots, \alpha_n)$,"* or, if it is from a sensing subsystem, we may say (somewhat anthropomorphically) it connotes the sensory information *"I see a wall with parameters $(\alpha_1, \ldots, \alpha_n)$."* (This difference highlights the need for an "interpretation function" in AI logical frameworks for sensory systems). The former interpretation we will call *absolute*; the latter we will call the *sensory* interpretation.[11] Note that we are *not* talking about the feasibility of building such a "perfect" sensing subsystem; we are discussing the meaning of the symbolic information passed between robot systems.

In light of the *sensory* interpretation above, consider the following. The sensory interpretation presumes some mechanism controlled by the robot, whose input is the world, and whose output is the symbolic piece of information $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$. Hence we may model this mechanism by some sensing map $\wp : C \times M \to \overline{B}$. We see that $\mathtt{wall}(\alpha_1, \ldots, \alpha_n) \in \overline{B}$, and that if $\wp(x, y) = \mathtt{wall}(\alpha_1, \ldots, \alpha_n)$, this means that when the robot is at configuration $x \in C$, then there is a wall with parameters $(\alpha_1, \ldots, \alpha_n)$ containing point y in the world M. The sensor space $\overline{B}$ then contains predicates such as $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$, and also some symbols like $\emptyset$ (meaning, "there is no wall at y visible from x", etc).

Now, the map $\wp$ defines a perfect virtual sensor—this sensor is analogous to an "oracle" in the theory of computation—it is a perfect information source that gives accurate "symbolic" information about the world. That is, $\wp$ takes $(x, y) \in C \times M$ as input, and returns perfect information about the existence of a wall at y. An alternative, equivalent interpretation

[11]This distinction is temporary; in our geometric framework there is no difference.

is to say that $\wp$ accesses a perfectly accurate geometric model, and "looks up" whether or not there is a wall at y visible from x. Again, we are not claiming that we can build such a sensor or a model; this discussion of oracle-like sensors is only for illustration. Once we understand what it means for a perfect sensor to output "symbolic" information about the world, then we can generalize this notion to an imperfect computational process that interrogates the world, and matches its (imperfect) observations to pre-encoded expectations about the structure of the world.

Now, consider the following alternative semantics for the statement

$$\wp(x, y) = \mathtt{wall}(\alpha_1, \ldots, \alpha_n). \tag{1}$$

We will stipulate that this statement encodes the following: *the sensor subsystem* $\wp$ *hypothesizes the existence of a wall containing* y*, with parameters* $(\alpha_1, \ldots, \alpha_n)$. That is, the subsystem has made observations of the world (through imperfect sensors), (possibly) retained state, and performed some computation. This computation relies on initial state, both in the form of a pre-existing robot program, and also in the form of pre-existing data-structures. Together, this initial state encodes expectations about the local fine-structure of the world—landmarks such as walls, corridors, hallways, etc. Let π_C be the natural projection of $C \times M$ onto C. The perceptual equivalence classes induced by $\wp$ (which we call *virtual* perceptual equivalence classes), are as follows: consider the inverse image of $\mathtt{wall}(\alpha_1, \ldots, \alpha_n)$, projected onto C—this is

$$\pi_C \wp^{-1}(\mathtt{wall}(\alpha_1, \ldots, \alpha_n)).$$

It consists of the set of configurations from which the existence of a wall with parameters $(\alpha_1, \ldots, \alpha_n)$ can be hypothesized. (Intuitively, the region from which such a wall can be "seen"). The boundary of this set contains the set of configurations touching the wall—this set is what we think of intuitively as the the "perceptual equivalence class *wall*."

Hence it is now clear how virtual sensors such as $\wp$ pre-encode expectations about the world. We model this pre-encoding as local sections of the world sensing bundle p (p corresponds to the "concrete" or actual sensors available to the robot). That is, corresponding to the value $\mathtt{wall}(\alpha_1, \ldots, \alpha_n) \in \overline{B}$, there is neighborhood U_t of B, and a local section $s_t : U \to C \times M$. This local section encodes how the sensor history (the image of b^*) is mapped into $C \times M$, to form the virtual perceptual equivalence classes of $\wp$. See fig. 9. Note that all such sections (eg., s_t) require *choices*, they are not canonical. These choices precisely encode our expectations about the local structure of the world. In mechanistic terms: "using" the sensor map p, we sense a series of values $b_1, b_2, \ldots \in B$. We find a subset U_t of B (or a series of subsets), containing these sensed values. We ask, what "features" in the environment could have caused these sensor values? This question is underdetermined, and, in any case, we usually do not have access to the inverse map p^{-1}. However, suppose we have some computer program—a computational process—that "matches" $b_1, b_2, \ldots$ against "expectations" about the world—in short, it produces a set S in $C \times M$. This set S represents a hypotheses—pairs $(x, y) \in C \times M$—that could have generated the sensor history $b_1, b_2, \ldots$ The process be be viewed as a local section of p with domain U_t. When projected onto C, the hypothesis S represents a virtual perceptual equivalence class. Note that it may be necessary for the computational process to have initial state, and to retain state (eg, previous sensor and control values) over time.

Hence, a key research challenge is to show how to construct virtual perceptual equivalence classes out of concrete perceptual equivalence classes. When recognizability is thusly constructive, we naturally obtain task-directed sensing strategies, driven by the information demands encoded in the structure of the recognizable sets. We have begun research to demonstrate this constructive program. For the interested reader, we sketch some of these constructions in appendix A.2.

9 Evaluation

We propose to develop and evaluate our theory by using it to perform manipulation experiments with mobile robots. These experiments will concentrate on planning and executing sensor-based manipulation strategies using our mobots. To accomplish this goal, we are using the theory above (secs. 2-8) in building mobile robotic systems (see [BD,RD]) to explore issues of robotic navigation, task-directed sensing, and active segmentation. We view *robotic navigation* as the science of building representations of the world inhabited by a mobile robot, in such a way as to allow the robot to move about its world reliably and to know its location in the world. *Task-directed sensing* allows the information demand for a particular task to drive the robot's sensing actions, thereby reducing the amount of sensing performed in the process of performing that task. *Segmentation* in vision and robotics is the separation of the world into discrete objects; in *active segmentation*, the robot segments its surroundings by physical interaction, e.g. possibly moving them, rather than passively looking at them.

Our mobot systems comprise a number of different techniques. We are developing algorithmic tactics to exploit sensory information, as described above. In addition to using previously developed sensors, we are designing new sensors which provide the robot with more appropriate sensory data (i.e., data which is in a form to be directly useful to the algorithms in use) [BD]. We are integrating hardware and software together to provide a stable platforms for experimentation on navigation, task-directed sensing, and active segmentation [RD]. Prototypes have been built for many of the subsystems of this platform, with integration into a full system ongoing [BD,RD]. We will continue work on these algorithmic and sensory subsystems, integrate them, and perform manipulation experiments [DJ4,BD].

We expect our work to impact the principles and practice of map-making for mobile robots. We view a *map* as a representation of a part of the world. In the context of robotics, a map is a representation which can be used to accomplish tasks. We view maps as and build them to be data structures which contain information about each part of the environment. A map is first treated as encoding geometric information. Maps can also store other kinds of information about the environment. For example, we annotate our maps with segmentation information ("this feature is part of object X; that other feature is part of object Y").[12] We can also annotate our maps with dynamical information by adding information

[12] X and Y are not "symbolic propositions", but refer to geometric constructions the robot has built to encode its observations about the world. The concept of "recognition" of objects X and Y seems clear to the robotics and vision community, but given our experiences at the 1st AAAI Fall Symposium on Sensory Aspects of Robotic Intelligence, we suspect there are lots of people who won't understand the difference between what we wrote above and a characteristic that makes systems flakey and failure-prone. Namely, that characteristic is the embedding of loads of semantic information in a representation, such as a

about the differential equations governing the movement of objects in the environment. The first experiments to perform will investigate the impact of our framework—particularly of perceptual equivalence classes, the lattice of recognizable sets, and the reachability graph of recognizable sets—on building, maintaining, and using these sorts of maps.

The next set of experiments will explore issues in task-directed sensing. We propose to use our framework of virtual perceptual equivalence classes (above) to design and build mobile robotic systems which will gather information about their environment in a task-directed fashion, as well as to perform active segmentation of the environment into separate features. We believe our framework will allow the robot to formulate experiments that it can perform on its environment to gain more information about the world; experiments can either be *exploratory* ("run and find out") or *dynamical* ("push something and see what happens").

Navigation for its own sake is a somewhat unsatisfying task for a mobile robot. In order for a mobile robot to be useful, it must be able to affect its work environment, through such actions as pushing, pulling, and grasping. We propose to use our framework for task-level planning and task-directed sensing, to identify and segment movable objects, and to guide plans that acquire, manipulate, and release objects. A scientific reason for examining tasks other than "pure" navigation concerns evaluation the robot's performance. It is often difficult to tell exactly why a navigation system exactly does what it does; "Navigate the environment" is a somewhat unsatisfactory goal in terms of evaluating success. If a robot navigates successfully from office A to office B, there exists the possibility that (for example) it got lucky with respect to its control uncertainty, and never wandered far enough from its originally planned path to need to use the more sophisticated parts of its navigation system. Often, even random actions may seem purposeful (see [E89]). By comparison, it is extremely unlikely that, for example, a robot will accidentally grasp five boxes in sequence and stack[13] them neatly down the hall, in a corner of Hopcroft's office. If a robot is given such an assignment and performs it successfully, then it is possible to say that the experiment was a success. For manipulation tasks, the goal is (often) a set of measure zero, and achieving it reduces entropy.

One of the chief difficulties in mobile robot manipulation is determining what parts of a sensed environment are part of the same object. A mobot whose task is to find canisters and remove them from a room, may not be able to perform segmentation using passive sensing (eg passive vision) alone. However, it can easily tell that a canister and (say) a desk are separate objects, if it pushes on the desk or the can and finds that it can push the one away from the other. This idea of *acting on the environment* to obtain information about it is akin to active sensing [WB]. In this particular instance, we are acting on the environment to determine connectivity of parts of the world—walls don't move when you push on them; boxes do; if the robot encounters four posts arranged in a square which move independently, then it probably has not found a chair; if they move rigidly together, then it is probably looking at the bottom part of a chair. This is more than simply analyzing connectivity; it is annotating

hypothetical vision system that returns information like "(chair, (pose($x, y, z, \rho, \theta, \psi$)))." The fact that this so-called "sensor" returns "chair" is totally bogus. Part of what we are saying is that black-boxes *can't* say things like that. Sensors say "intensity b at (x, y)" and eventually (hopefully) the robot might conclude "I think there's an object that fits in to the class of objects perceptually equivalent to what I have been told to call a 'chair' at location (x, y, θ) relative to me, give or take $(\epsilon_x, \epsilon_y, \epsilon_\theta)$." See [DJ2] for more details.

[13]We will explore planar stacking.

the geometric map with information obtained through *experimental dynamics*. We believe our framework provides a theory for how to drive the experments in active segmentation and experimental dynamics using task-directed sensing (as described in secs. 2-8). Furthermore, we feel that we can develop this theory and use it to perform experiments on sensor-based manipulation strategies for mobile robots in uncertain environments. Manipulation strategies seem a good domain because they exercise many aspects of our framework for task-level planning and task-directed sensing. Second, there are concrete intermediate goals along the way, such as map-making, active segmentation, and experimental dynamics. Finally, we believe it is relatively easy to evaluate the success of such sensor-based manipulation strategies, and hence it is possible to have objective performance criteria by which to judge progress.

9.1 Conclusion

In this paper, we considered how a robot may interpret its noisy, sparse sensors, and direct its actions so as to gain sufficient information about the world to achieve a task. Sensing forces us to view the world as a projection, and this projection divides the world into perceptual equivalence classes. We investigated how to model various sensors, including range-finders, cameras, and odometry. We build new sensing maps from old ones, creating a hierarchy of virtual sensors. These constructions are guided by the information needs of the task.

We developed tools, including virtual sensors, the recognizability gradient, and the perceptual limit, which permit a precise characterization of the information flow into and through the robot. The robot's capabilities may be explored, as well as fragilities in robot plans or programs, due to problems in matching expectations against observations.

A critical property of our framework is that the data structures and representations employed are *a priori* recognizable features, or landmarks, in the world. Our theory is derived from a "calculus of the observable", and when recognizability is thusly constructive, we naturally obtain task-directed sensing strategies, driven by the information demands encoded in the structure of the recognizable sets.

One advantage of this formulation may be that it defines a set of experiments that the robot may perform to try to recognize a virtual perceptual equivalence class. That is, this construction of virtual perceptual equivalence classes explains how the observed sensor values should change with respect to strategies the robot can perform. By performing these actions and observing its sensors, the robot may verify hypotheses about the existence and geometry of the virtual perceptual equivalence classes it must recognize in order to perform a task. Specifically, we may regard certain sensors as "concrete" sensors, relative to a high-level "virtual" sensor. The resulting "concrete" perceptual equivalence classes are not likely to be human-readable in general. More likely, a robot task might be specified in terms of walls, corridors, and hallways (for example: "Go down the hall, take the second right, and follow the wall as it curves around until you reach the atrium.") These are the virtual perceptual equivalence classes with respect to which a human might wish to express a task. But how can the robot recognize these classes, using the sensors and computation it carries on-board? Instead of trying to provide the robot with "perfect sensors" such as a "wall oracle", a "hallway oracle", etc., instead, we can tell it how to recognize virtual perceptual equivalence classes from concrete ones. Using this method, the robot can then perform experiments to verify which virtual perceptual equivalence classes it is in. This is the process we call *task-directed sensing.*

APPENDICES

The research agenda in secs. 2-8 is self-contained, in that the basic ideas behind our paper are accessible without reading the appendices. However, experienced readers may have questions that we remain unaswered, for example, *How would a more complicated example work?* and *Can you say more on how sensor noise and uncertainty are to be handled?* While the answers are implicit in our previous work, for the interested reader we elaborate on the two lacunæ above, in appendices A and B.

A Example: Modeling Sensors

We now proceed to explore just how physical sensors are modeled, and we introduce the concept of "virtual sensors".

Consider a range-finding sensor such as an ultrasonic sonar transducer (this discussion would also apply to a laser range finder). The hardware that supports these transducers has been designed to provide time-of-flight information necessary to determine how long it took a sonar ping to travel to an object and back again. If we make an assumption about the speed of sound in air, we an obtain approximate distance information given the time of flight.

Hence the physical range-finding sensor returns a value b_p which is an integer—say, the time of flight measurement to the resolution permitted by the hardware timer on a mobile robot. It is also possible to get no return. Hence, we can model the hardware as returning a value b_p in a physical sensor space B_p, where $B_p = Z \cup \{\infty\}$, where ∞ means "no return."

We do not, however, choose to model the range finder as a mapping from controls to B_p. That is because we wish to define a sensing map that encodes the *information* obtained in the return ping. The information contained in a (non-infinite) return ping is not simply "There exists an obstacle at distance d." It is much more:

> *(*) "There exist no obstacles up to distance d; there exists an obstacle at distance d; and beyond d we have no information."*

Consider our simplified model of a range finder, $p : C \times M \rightarrow B$. Let us include the sensor controls $\theta \in S_1$ as a factor of C so that the local sensor map is simply $p_x : M \rightarrow B$. We developed this sensor as follows. First we asked the question:

> *(Q1) Given a point y in the world M, what effect does y have on the values we sense?*

We call the answer to this question the *pointwise* definition of the sensing map: it gives us the rule $p_x(y) = b$. Now, using direct sums it is easy to say what sets of values we get when we apply p_x to a *set* S of points in the world:

$$p_x(S) = \bigcup_{y \in S} \{\, p_x(y) \,\}. \tag{2}$$

For example, suppose we apply p_x to a ray $S_{x,\theta}$ originating at x and pointing in direction θ. Suppose the first object that the ray intersects is at distance d from x. Then clearly we have

$$p_x(S_{x,\theta}) = \{\, \emptyset, \lfloor d \rfloor, \infty \,\}. \tag{3}$$

Note that eq. (3) precisely encodes the information in statement ($*$) above. To summarize: we have a physical sensor whose domain is B_p. It encodes the actual values returned as signals or symbols from the hardware. We model the information contained in these signals by a pointwise map such as p_x (the range sensor) whose domain is B. The behavior of this map on sets defines the information content of the sensor. Note that this information is captured by (a) the set-valued map $p_x : 2^M \to 2^B$, (b), an "accumulation" or "interpretation" function on sets of values, that tells us that eq. (3) *means* the paragraph ($*$). It may be useful to think in terms of programming language concepts: (a) corresponds to a `map`[14] of p_x over S; (b) corresponds to `accumulate` applied to the result of `map`.

Finally, we note that B_p may be viewed as a *projection* of B. In essence, the sensor map p_x may be viewed as an *virtual* sensor that encodes the information available in the physical sensor whose range is B_p. The process of interpreting the physical sensor may be viewed then as a lifting of the physical sensor to p_x:

$$\begin{array}{ccc} & & B \\ & \overset{p_x}{\nearrow} & \downarrow \pi \\ M & \longrightarrow & B_p. \end{array} \tag{4}$$

We present models for other types of sensors, such as cameras, in section A.1. In that section we also explore odometry as a sensor, and work through a more detailed virtual sensor example. We close with one last didactic example:

Construction A.1 *The sensor "value" b could represent a probability distribution as in [ME, Mor]. In this case, the information content of the sensor would still be ($*$). We construct a virtual sensor $\mathcal{P}_x$ that takes the output of p_x as its input. The virtual sensor $\mathcal{P}_x$ works as follows. Let S be a subset of M, as above. Given the information ($*$), which is encoded as the set $p_x(S)$, $\mathcal{P}_x$ treats this information as its only prior, and constructs a distribution. Hence, $\mathcal{P}_x(S)$ will be a probability distribution over S that represents the probability, given ($*$), of the existence of obstacles at any point y in S. That is, the distribution $\mathcal{P}_x(S)$ maps points y in S to the probability of y being inside an obstacle. Now, in general, this model requires the domain of $\mathcal{P}_x$ to be a function space—all possibile probability distributions over all subsets of M. However, in practice, B would be a simple parametric family of distributions. The distributions used in practice are very simple, and are parameterized by the values of p_x. This example illustrates the power of virtual sensors. See sec. A.2 for more details on virtual sensors.*

A.1 Example: Modeling Complex Sensors

In this section we discuss how to model some types of sensors, particularly, a camera, odometry, and "virtual sensors". The last can be useful in the high-level programming of autonomous agents.

[14] `map` in SCHEME and ML; `mapcar` in COMMONLISP.

A.1.1 Example: Modeling a Camera

Suppose we have a 1 pixel greylevel camera. Suppose there are 8 bits of greylevel and so we may imagine the "value" of this sensor as lying in Z_7, the integers 0 through 7. Suppose the camera is situated and pointed as determined by its configuration x in sensor configuration space C, we now ask question (Q1) above. The answer to (Q1) is: the camera essentially measures luminosity, and hence a point y in the world can cause the camera to "read" between 0 and 7. ("Transparent" and occluded points, of course, will read zero also).

We now set about to model the information available in this sensor. First we define the following function on any set S in the world M:

$$\mathtt{map}(p_x)(S) = \bigcup_{y \in S} \{ y \} \times \{ p_x(y) \}. \tag{5}$$

Thus, the result of (5) on a set is a "table" of what luminosity values are assigned to each point y in S. A little thought shows that this "table" is exactly a *function*, namely the restriction of p_x to S:

$$\mathtt{map}(p_x)(S) = p_{x|_S}. \tag{6}$$

Next, we have to add up all these luminosity values to obtain the value representing the camera signal. That is, the value of the sensor on a set S is not $p_{x|_S}$, but rather the sum[15] $\sum_{y \in S} p_x(s)$. More generally the sum is infinite and of course we write this as an integral:

$$\mathtt{Accumulate}(p_{x|_S}) = \int_S p_{x|_S}. \tag{7}$$

We summarize as follows. To define the sensor map p_x that models the information content of a 1-pixel camera sensor, we first define p_x pointwise, as in (Q1) above. In this case, the physical sensor is identical to the virtual sensor corresponding to the information content of the camera "signal." Next, we wish to define the behavior of the virtual sensor on sets S in the world. To model the information obtained in "applying" p_x to a set, we define a sensor map P_x on sets of M. P_x is the composition of `map` and `Accumulate`. More precisely:

$$P_x = \mathtt{map}(p_x) \circ \mathtt{Accumulate}.$$

We observe that `map` is essentially the restriction map, that restricts p_x to a subset S. The set S is analogous to the notion of *support* that is common in mathematics and computer vision. `Accumulate`, we have seen, is simply integration, and hence $P_x(S)$ is simply integration of the function p_x over its region of support:

$$P_x(S) = \int_S p_x.$$

If we restrict the sets S to be open, then all the open sets of M form a lattice. In mathematics, a function such as $\mathtt{map}(p_x)$ that assigns, to each open set S, the restriction

[14] This example arose in discussions with Matt Mason, and we are very grateful to him for his suggestions and insight in developing it.

[15] One technical point is that this sum saturates and Z_7 is not really the ring of integers mod 7.

map $p_{x|_S}$, is called a *sheaf.* The `Accumulate` function is a kind of *evaluation* on the sheaf. Hence, cameras and similar sensors are naturally modeled as sheaves on manifolds. There is a wealth of literature and results on sheaves.

Finally, note that a multi-pixel camera can be modeled as product of many p_x-type maps, i.e., as a product of sheaves.

A.1.2 Example: Odometry Sensor

As a further example, we explore how to model an odometry sensor. Formalizing this example appears somewhat technical, and we present the details in sec. A.1.2. Now, we just give the basic idea. Consider the notion of *forward-projection* as formalized in, for example, [Erd]. In a configuration space C, let R be a set of initial conditions; that is, the initial state of the robot system is known to be some point in R. let us say this initial state takes place at time $t = 0$. A choice of actions, or controls, which we will call θ, determines the evolution of the system for times $t > 0$. θ is a quite arbitrary index and could range over feedback loops and complex strategies (for example); in [LMT, Erd] it is intended to index the "angular direction" of a commanded generalized damper motion (which of course, is a feedback loop). Erdmann defines the *forward projection* $F_\theta(R, t)$ to be the subset of configuration space that is reachable at time t from initial conditions R and under controls θ [Erd]. That is, the forward projection is a bound on the predictive forward-simulation accuracy of the robot planner; it encodes the robot's knowledge of the effect of executing action θ from initial conditions R. If we view the robot as a dynamical system subject to uncertainty, then the forward projection is the set resulting from integrating the differential inclusion that models the system; see [AC] for example.

It is clear that the forward projection (and its suitable restrictions) model the information available in an odometry sensor or by dead-reckoning computation on control history (integration of commanded controls with error bounds). The way a forward projection is "constructed" via the operations of integration and union is fairly clear: first, we integrate for all valid controls in the differential inclusion to obtain a family of trajectories. Next, we take the union of the states in the image of these trajectories. Indeed, this integration and union construction is explicit in [Erd]. As above (Sec. A.1.1), it is possible to represent this operation as a `map` operation composed with an `Accumulate` operation. See below, sec. A.1.2.

We now describe briefly how the odometry sensor may be viewed in our framework. The knowledge available to the system consists in knowing R, t, and θ. Hence, we may view the tuple (R, t, θ) as a "value" $b^\diamond$ in a sensor space $B^\diamond$. $B^\diamond$ in turn is the domain of a sensing map $p^\diamond$, and we define

$$\pi_C p^{\diamond-1}(b^\diamond) = \pi_C p^{\diamond-1}((R, t, \theta)) = F_\theta(R, t). \tag{8}$$

Details of the Odometry Sensor

Suppose we have a robot with perfect velocity control, and we have perfect knowledge of the commanded velocity $\dot{x}(t)$ as a function of time. In short, our system is described by a differential equation

$$\dot{x} = f(x,t); \tag{9}$$

the state x can be found by integrating this system. Now, we suppose that "knowing" our controls can be modeled by a "sensor" v_x^* whose value is $\dot{x}$. Then, a perfect odometry sensor x^* can be constructed by integration. That is, we have

$$p_x(t) = \int_0^t v_x^* = \int_0^t \dot{x}(t)dt.$$

More formally: let T denote time (the non-negative real numbers), and $\Phi : C \times T \to C$ be the flow on C generated by f, i.e., by eq. (9). Let $\phi_{x_0} : T \to C$ be an integral curve of f with initial conditions x_0, so $\phi_{x_0}(t) = \Phi(x_0, t)$. We define $x^*(t) = \phi_{x_0}(t)$.

If we have imperfect control we must replace the differential equation $\dot{x} = f(x,t)$ by a differential inclusion $\dot{x} \in V_\theta(x)$, where $V_\theta(x,t)$ is a set of tangent vectors; for example, it might be a cone of vectors in direction θ. In general, V_θ is the set-valued map that associates to the state x of the system the set of feasible velocities (see [AC]). In this case we can define an approximate odometry sensor by integration and union:

$$x^*(t) \in F_\theta(t) = \bigcup_{v* \in V_\theta} \int_0^t v^*(t)dt.$$

More generally, following [Erd], we define the forward projection $F_\theta(R,t)$ as above (sec. A.1.2). Then we parameterize the flow Φ by the controls v^* to obtain $\Phi_{v^*} : C \times T \to C$. Then the construction of the forward projection via integration and union is clear:

$$F_\theta(R,t) = \bigcup_{\substack{v* \in V_\theta \\ x_0 \in R}} \Phi_{v^*}(x_0, t).$$

Using eq. (8) to define the sensing map $p^\diamond$ for odometry or dead-reckoning, it is then clear how the sensing map $p^\diamond$ can be constructed via an appropriate `map` and `Accumulate` of the control and position sensor maps v^* and x^*.

A.2 Example: Virtual Sensors

We now work a brief example on the construction of virtual sensors. See fig. 8. We consider three sensing maps: the first, $p^\diamond$, is the sensing map for odometry described above (eq. (8)). The second is the range-finder sensing map p_x from section 5 above. As usual we globalize p_x. Let us call the resulting globalization p^* (instead of p), which is defined as follows:[16]

[16]Do not be disturbed that the robot both controls and "senses" θ in example A.2 (above). We could modify the example to model a "rotating range-finder" like a sweep sonar. Such a device consists of a linear range-finder on a constantly rotating platform. In such a device, θ is not servoable, but it may be sensed. The device returns a value of the form $(\theta, p_x(\theta, y))$. Hence, it can be modeled by a sensor map of the form

$$\begin{array}{rccl} p_x^+ & : & M & \to & B^* \\ & & y & \mapsto & (\theta, p_x(\theta, y)). \end{array} \tag{10}$$

Construction A.2 *We construct the map p_x^*. Let B^* be the larger sensor space $B^* = S^1 \times B$ where B is as in section 5. Then we define p_x^* to be "$\mathrm{id}_{S^1} \times p_x$" where* id *is the identity on S^1. That is*

$$p_x^* \; : \; \begin{array}{ccc} S^1 \times M & \to & B^* \\ (\theta, y) & \mapsto & (\theta, p_x(\theta, y)). \end{array} \tag{11}$$

It is easy to see that each perceptual equivalence class in fig. 6 is of the form $\pi_M p_x^{*-1}(b)$ for $b \in B^*$.

We demonstrate how the perceptual equivalence classes for p^* may be constructed using

1. The raw sensor values of the odometry sensor $p^\diamond$,
2. The raw sensor values of the range sensor p^*, and
3. Integration and union of the perceptual equivalence classes from $p^\diamond$.

That is, without knowing an explicit inverse for p^*, but knowing (i) the perceptual equivalence classes induced by $p^\diamond$, and (ii) the values of both $p^\diamond$ and p^*, how can we hypothesize the existence and geometry of the perceptual equivalence classes induced by p^*? Here is our general method of attack. Since we know the perceptual equivalence classes induced by $p^\diamond$, let is call it a *concrete* sensor, and its perceptual equivalence classes *concrete* perceptual equivalence classes. Since we do not know (*a priori*) the perceptual equivalence classes induced by p^*, let us call p^* a *virtual* sensor, and the perceptual equivalence classes induced by p^* *virtual* perceptual equivalence classes. Our question may then be concisely phrased as:

> *How can we hypothesize the existence and geometry of the virtual perceptual equivalence classes from the concrete perceptual equivalence classes we know?*

In general, this problem, which is key to constructing one sensing map from another, can be rather difficult. It is necessary to specify how virtual perceptual equivalence classes are constructed out of (or from) concrete perceptual equivalence classes. That is, it seems necessary for us to tell the robot "What would virtual class $\pi_C p^*(b^*)$ 'look like' through $p^\diamond$?" There is a general answer:

$$p^\diamond(p^{*-1}(b^*));$$

however, this characterization does not directly suggest a computational solution.

However, here are some operational ways of defining virtual perceptual equivalence classes. These techniques require the ability to measure change in the sensing map p^*, and to union "according" to that change. It helps to think of this "controlled union" as a kind of "integration."

Now, for any sensing map, we define its "recognizability gradient:"

An alternative to a "sweep" model is to assume a ring of sensors giving nearly continuous coverage, as often sported by mobile robots.

In addition, the sensor controls (in the case of p_x, S^1) may be incorporated into the configuration x of the robot. In that case, the sensor "aiming" can be viewed as a kind of control parameter that we command and sense. In this case, control and sensing of θ are perfect, but this need not be the case.

Definition A.3 *Let $p : C \times M \to B$ be a differentiable sensing map. Define $\nabla_C p$, the* recognizability gradient *of p to be the projection of the gradient ∇p of p onto TC. Consider the natural tangent space projection $T\pi_C : T(C \times M) \to TC$. Then we have*

$$\nabla_C p = T\pi_C \nabla p.$$

In the case where p is discrete or non-smooth, we will approximate the recognizability gradient by finite differences; we assume this is possible for the maps $p^\diamond$ and p^*. Now, consider the following definitions for virtual perceptual equivalence classes:

1. A virtual perceptual equivalence class can be defined as the union of a set of contiguous concrete perceptual equivalence classes (of $p^\diamond$) with the same recognizability gradient $\nabla_C p^*$.

   (a) More specifically, a virtual perceptual equivalence class can be defined as the union of a set of contiguous concrete perceptual equivalence classes (of $p^\diamond$) with the *zero* recognizability gradient ($\nabla_C p^* = 0$). This implies that all these (concrete) classes have the same value with respect to p^*.
   This construction allows us to use odometry to "trace out" regions of constant range-finder values. These regions will be (eg) strips parallel to linear walls at a constant distance. The width of the strips will be determined by the resolution of the range-finder's distance measurement.

2. Consider a "sequence" of concrete perceptual equivalence classes (of $p^\diamond$)—eg, a linear ordering along some path in C. Suppose as we traverse this sequence, we move in the direction of the recognizability gradient $\nabla_C p^*$. We can define a virtual perceptual equivalence class that is the *limit* of this process.

   (a) More specifically, we have:

   Construction A.4 (∗) *Given a sequence of concrete perceptual equivalence classes $\mathcal{S} = \{S_1, \ldots, S_i, \ldots\}$ where $S_i = \pi_C p^{\diamond -1}(b_i^\diamond)$ for some $b_i^\diamond \in B^\diamond$. Let I^- denote the half-open unit interval $[0, 1)$, and consider any smooth path $\phi : I^- \to C$ such that $p^\diamond(\phi(I^-)) = \mathcal{S}$. Suppose further that at all times t in I^- we have the path ϕ pointing in the direction of the recognizability gradient $\nabla_C p^*$:*

   $$\nabla_C \, p^*(\phi(t)) = \dot{\phi}(t). \tag{12}$$

   Then we may define a virtual *perceptual equivalence class $\overline{S}$ which is the* limit *of the sequence $\mathcal{S}$:*

   $$\overline{S} = \lim_{\nabla_C p^*} S_i, \tag{13}$$

   and $\overline{S}$ contains $\phi(1)$.

Now, let $b^* \in B^*$, the sensor space of p^* (recall section 5 and construction A.2, above). It is now clear that the perceptual equivalence classes of p^* can be defined using the "zero recognizability gradient" construction (1a, above). Specifically:

- A virtual perceptual equivalence class $\pi_C p^*(b^*)$ can be defined as the union of a set of contiguous concrete perceptual equivalence classes $\{\pi_C p^{\circ -1}(b_i^\circ)\}$ of p°, such that all these (concrete) classes have the same value b^* under p^*.

Perhaps the most interesting virtual sensors we can construct entail no additional raw sensors, but only computation and history. Consider a "virtual wall sensor." Suppose that x is a configuration in C, and a point y on M lies on a wall that has absolute[17] orientation θ, length l, and distance d. It might be convenient to have a "wall recognition sensor" that would tell us about the wall at y when we are at x. We imagine this special *virtual* sensor as a new sensing function $\wp$. In the case of the particular x and y above, we want $\wp$ to return the "value" $\mathtt{wall}(\theta, l, d)$. Think of this "value" as a "symbolic expression" whose "meaning" is that there is a wall, containing y, with absolute orientation θ, length l, and distance d. We might imagine defining $\wp$ as follows.

$$\begin{aligned} \wp \ : \ C \times M &\rightarrow B_w \\ (x,y) &\mapsto \begin{cases} \mathtt{wall}(\theta, l, d) & \text{if there is a wall at } y\text{, and } y \text{ is visible from } x; \\ \emptyset & \text{otherwise.} \end{cases} \end{aligned} \tag{14}$$

Note that all we have done is specify what $\wp$ should do; that is, we have written down a specification for a sensor we might like to have. Eq. (14) gives us no clue how to implement this specification using computations on our other sensors. What to do in general with such a specification eludes us. In this case, however, the answer is clear:

Consider p^*, the range sensor, as a concrete sensor, and view the perceptual equivalence classes induced by p^* as the concrete perceptual equivalence classes. We will define the virtual sensor $\wp$ relative to the concrete sensor p^*.

The non-trivial perceptual equivalence classes induced by $\wp$ are generated by sets of pairs (x,y) in $C \times M$, such that there is a wall at y and it is visible from x. Each of these virtual perceptual equivalence classes is simply constructed as the union of a set of contiguous concrete perceptual equivalence classes (of p^*) with the same recognizability gradient $\nabla_C p^*$. That is, each concrete perceptual equivalence class $\pi_C p^{*-1}(b^*)$ may be envisioned as a "strip" parallel to a linear wall. A set of these strips with the same recognizability gradient is a set of perceptual equivalence classes all "caused" by sensing the same wall. These strips have the same recognizability gradient $\nabla_C p^* = const$, since, in any one strip, moving along, away, or toward the wall will cause the same changes (respectively) in the sensing map p^*. This is just a way of saying that making incremental motions in C results in the same observed changes in the sensing map p^*; i.e., these "strips" $\pi_C p^{*-1}(b^*)$ have the same recognizability gradient. Hence, they may be unioned into a larger set, namely the class of configurations that can "see" the same wall. For $b_w \in B_w$, this set $\pi_C \wp^{-1}(b_w)$ is defined to be the union of all contiguous strips of the form $\pi_C p^{*-1}(b_i^*)$ with a fixed, constant recognizability gradient $\nabla_C p^*$.

One advantage of this formulation may be that it defines a set of experiments—incremental motions in C that the robot may perform to try to recognize a virtual perceptual equivalence class such as $\pi_C \wp^{-1}(b_w)$. That is, this construction of virtual perceptual equivalence

[17] We could also define the orientation and distance to be relative to x, but the construction is a bit more complicated.

classes explains how the observed sensor values should change with respect to incremental motions the robot can perform. By performing these actions and observing its sensors, the robot may verify hypotheses about the existence and geometry of the virtual perceptual equivalence classes it need recognize in order to perform a task. Specifically, we may regard the sensors $p^\diamond$ and p^* as "concrete" sensors, relative to the wall sensor $\wp$. The resulting "concrete" perceptual equivalence classes are not likely to be human-readable in general. More likely, a robot task might be specified in terms of walls, corridors, and hallways (for example: "Go down the hall, take the second right, and follow the wall as it curves around until you reach the atrium.") These are the virtual perceptual equivalence classes with respect to which a human might wish to express a task. But how can the robot recognize these classes, using the sensors and computation it carries on-board? Instead of trying to provide the robot with "perfect sensors" such as a "wall oracle", a "hallway oracle", etc., instead, we can tell it how to recognize virtual perceptual equivalence classes from concrete ones. Using this method, the robot can then perform experiments to verify which virtual perceptual equivalence classes it is in. This is the process we call *task-directed sensing.*

The construction above defines a somewhat peculiar virtual perceptual equivalence class–the set of configurations that can see a wall. Note that we can also define the more intuitive set, the class of configurations that are "against the wall." This may be done as follows. We use construction A.4 above. We define the virtual perceptual equivalence class

$$\overline{S} = \pi_C \wp^{-1}\Big(\texttt{on-wall}(\theta, l, d)\Big) \tag{15}$$

as the *limit* of a sequence of strips $\pi_C p^{*-1}(b_i^*)$, such that the recognizability gradient $\nabla_C p^*$ decreases as the sequence is traversed. This decrease, of course, corresponds to the rangefinder's distance readings going down as we get close to the wall. We simply mimic the construction A.4 above. The basic idea of this construction is easy. Recall that, for $b_w \in B_w$, the set $\pi_C \wp^{-1}(b_w)$ is defined to be the union of all contiguous strips $\pi_C p^{*-1}(b_i^*)$ with constant recognizability gradient $\nabla_C p^*$. $\pi_C \wp^{-1}(b_w)$ is the set of configurations that can see the wall specified by $b_w = \texttt{wall}(\theta, l, d)$. This entire region has a non-zero constant recognizability gradient $\nabla_C p^*$. We imagine this constant gradient inducing a flow on the concrete perceptual equivalence classes $\{\, \pi_C p^{*-1}(b_i^*) \,\}$ constituting the virtual perceptual equivalence class $\pi_C \wp^{-1}(b_w)$. We follow this flow to its limit, and that limit is the set of configurations that touch the wall b_w.

We now give the formal construction. One minor point is we must flip the sign of the recognizability gradient, since we want to converge *to* the wall, not away from it, and the distance values, of course, decrease to zero as we approach the wall.

Construction A.5 *Suppose we are given a sequence of concrete perceptual equivalence classes $\mathcal{S} = \{\, S_1, \ldots, S_i, \ldots \}$ where $S_i = \pi_C p^{*-1}(b_i^*)$ for some $b_i^* \in B^*$. Let I^- denote the half-open unit interval $[0,1)$, and consider any smooth path $\phi : I^- \to C$ such that $p^*(\phi(I^-)) = \mathcal{S}$. Suppose further that at all times t in I^- we have the path ϕ pointing in the direction of the recognizability gradient $-\nabla_C\, p^*$:*

$$-\nabla_C p^*(\phi(t)) = \dot{\phi}(t). \tag{16}$$

Then we may define a virtual *perceptual equivalence class* $\overline{S}$*, corresponding to eq. (15) which is the* limit *of the sequence* S*:*

$$\overline{S} = \lim_{-\nabla_C p^*} S_i, \tag{17}$$

and $\overline{S}$ *contains* $\phi(1)$.

Finally, we note that if we replace our range sensor with the sensing map described above in construction A.2, that our analysis and definitions may be suitably generalized. For reasons of space, we feel we must leave details to the interested reader.

B Discussion: Effects of Sensor Noise

We now take a step back, and address a concern that arose in section 5 when we declared our example range sensor to be a perfect one; i.e. no noise or uncertainty was modeled. It is clear that coping with uncertainty is critical, and in fact all of the constructions we describe work for sensor models with uncertainty. We explore how this is done here.

Sensor noise can be modeled by passing to set-valued functions. That is, we consider $p_\phi(y)$ to be a subset of B, representing a bound on all possible sensor values. In general, this can destroy the transitivity of perceptual equivalence classes which are defined in terms of inverse images of points in the sensor space B under sensing maps as in, for example, def. 5.1 (above). However, the reflexivity and symmetry are retained. Hence a definition of a perceptual equivalence class becomes analogous to an "adjacency" relation, and its transitive closure analogous to "connectedness". This issue is subtle. We now investigate it carefully.

B.1 Noise as a Set-valued World-Sensing Map

First, we show how sensor noise can destroy transitivity. To do so we develop a world-sensing bundle which models an LMT-like position sensor (with noise) (see [LMT]).[18] Most of the points in this section (B.1) are either definitional or quite clear from the development in [LMT,E], but since they are important, we include the discussion for completeness.

Let C, and M both be the plane $\Re^2$. Define a perfect position sensor to be the raw sensing map $p^* : C \times M \to \Re^2$ that sends (ϕ, y) to ϕ. We think of this map as "sensing" the configuration ϕ of the system.

Now, let us introduce noise into the model. Under an LMT-like model of error, we are only guaranteed that the image of ϕ under p^* will lie "nearby", eg in some ball of radius ϵ about ϕ. Hence we have $p^*(\phi) \in B_\epsilon(\phi)$. We can chose to model the noise sensor p^* as a nondeterministic sensor, or as a set-valued function. Let us choose the latter. We make the image of p^* be the powerset of $\Re^2$. The value of p^* represents a bound on all possible sensor values. We know by inspection that the value of p^* will in fact always be a disc in the plane. Suppose we try to generalize our notion of perceptual equivalence to set-valued maps. Consider the following

[18]This example arose in discussions with M. Erdmann, and we are grateful to him for his suggestions.

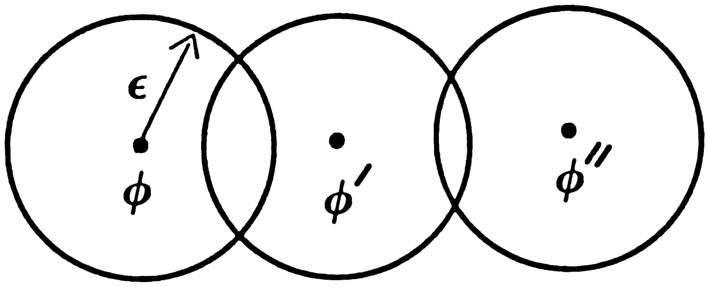

Figure 10: Three configurations with ϵ-balls around them. Under the definition in idea B.2, ϕ is "perceptually equivalent" to ϕ' and ϕ' is "perceptually equivalent" to ϕ'', but ϕ is *not* "perceptually equivalent" to ϕ''. Hence the relationship "perceptually equivalence' is reflexive and symmetric, but not transitive.

Idea B.1 (*) *We could define the two configurations ϕ and $\phi' \in C$ to be* perceptually equivalent *if there exist y and $y' \in M$ such that*

$$p^*_\phi(y) = p^*_{\phi'}(y'). \tag{18}$$

This definition is clearly inadequate, since the value of p^* is always a disc of radius ϵ, and these two discs are identical only when $\phi = \phi'$. Suppose we try:

Idea B.2 (*) *Modify eq. (18) in idea B.1 to be:*

$$p^*_\phi(y) \cap p^*_{\phi'}(y') \neq \emptyset. \tag{19}$$

This is better, since it means that ϕ can "look like" ϕ' when they are confusable[19]–that is, when there is some possibility that the sensor p^* will return the same position value from ϕ as from ϕ'. The problem with declaring ϕ and ϕ' to be "perceptually equivalent" is that the relationship we have defined is not an equivalence relation, because it it not transitive: see fig. 10.

The relationship we obtain is reflexive and symmetric, but not transitive. Hence it is analogous to the "adjacency" relationship in (eg) graph theory or topology. One idea is to construct an equivalence relation by taking the transitive closure. The motivation for this is that the transitive closure of "adjacent" is "connected," which is a useful notion. However, in our example of p^*, this results in identifying the entire plane into one equivalence class.

One could try to get around this problem by modeling sensor noise by a non-deterministic sensing map. However, the two constructions are basically equivalent. We can see this as

[19]See [Buc, LMT, E, Don], where the concept of "confusability" was formalized in an LMT context.

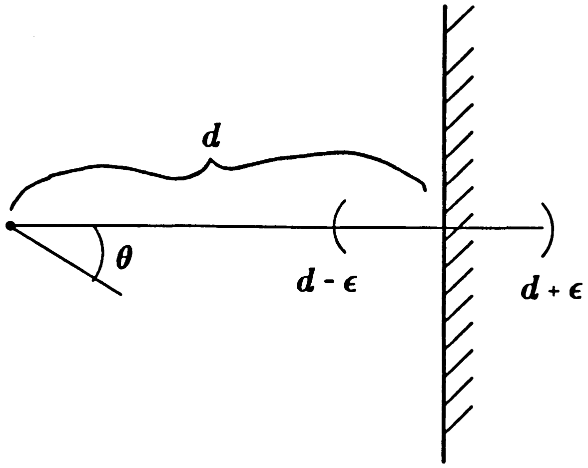

Figure 11: The robot is at point x, with the range-finder aimed in direction θ. A wall at point y is at distance d from the robot; due to uncertainty in range-finding sensor, the range value sensed is only guaranteed to be inside $(d-\epsilon, d+\epsilon)$.

---

follows. We notice that we can restrict to domain of p^* to obtain

$$p^* : C \times M \rightarrow \{\, \epsilon\text{-balls in the plane}\, \}.$$

The set of all ϵ-balls is parameterized by their centers, hence we can use $\Re^2$ as the domain of p^*. Now, there are two interpretations possible for p^*. The first is the same as the perfect sensor, since it sends ϕ to the ball centered on ϕ. A second interpretation is as a non-deterministic map that takes ϕ to some sensed value $s \in B_\epsilon(\phi)$. Now, the set of configurations consistent with s (i.e., that could have given rise to the sensed value s) is the ball $B_\epsilon(s)$ about s. Hence a non-deterministic sensor is a map $\phi \mapsto s$; but the set of consistent interpretations of s is a ϵ-ball about s. Hence the non-deterministic sensor may be naturally viewed as a set-valued map $\phi \mapsto B_\epsilon(s)$.

B.2 Perceptual Aliases

Given the discussion above, the situation may look a bit grim for a calculus of the observable based on perceptual equivalence. However, there is a solution, which we give now. We show how noise can be modeled within our framework, while preserving the structure and flavor of all we have developed. The basic idea behind this construction is to turn a weakness into an advantage, and exploit the ability of our sensor model to encode non-determinism (via hidden state) within a deterministic model. Once again, a more detailed study can be found in [DJ2].

As an example, consider once more our range finding sensor, as described in section 5 above. A "picture" of this sensor model is shown in fig. 11. The sensor configuration is the point $\phi \in C$. In particular, $\phi = (x, \theta)$ as shown. The point $y \in M$ is on a wall. For a perfect sensor, we have $p(\phi, y) = \lfloor d \rfloor$ where d is the distance from x to y. However, for a noisy sensor, we might have the following sensor model. There is some error, say ϵ, which lies in an interval U. This error varies non-deterministically. The error causes the sensor to return only approximate information, and sometimes to return $\emptyset$. We can think of this error "signal" ϵ as fluctuating non-deterministically, and "corrupting" the perfect sensor's readings. This results in an imperfect, or noisy sensor, which we might call p^*. See fig. 11. The effect of p^*, when applied to the point y shown, is to return some distance in the interval $[d - \epsilon_{max}, d + \epsilon_{max}]$ (or $\emptyset$). Now, for a sensor with infinite resolution (no discretization), we would have $p(\phi, y) = d$ and

$$p^*(\phi, y) = p(\phi, y) + e(\epsilon) = d + e(\epsilon).$$

However, for a sensor with limited resolution, the situation is more complicated. For such a sensor p^*, (for the same point y as above):

$$p^*(\phi, y) = \begin{cases} \emptyset & \text{if } |\epsilon| > \epsilon_{max} \text{ ;} \\ \lfloor d + e(\epsilon) \rfloor & \text{otherwise.} \end{cases} \tag{20}$$

In eq. (20), we assume that $e(\epsilon)$ is some simple function of the error ϵ. In this example, we will assume e is the identity. All our results go through when e is merely injective.

Our idea is to model the "error space" U as part of the world. We regard the "new world" M' as incorporating both M and the error space U, so $M' = M \times U$. We define a new sensing map $\underline{\mathbf{p}} : C \times M' \to B$, that is,

$$\underline{\mathbf{p}} \ : \ C \times M \times U \ \rightarrow \ B \tag{21}$$

where $\underline{\mathbf{p}}(\phi, y, \epsilon)$ is given by eq. (20). That is, the state of the world contains U, which determines the error in the sensor. This process is quite general, and extends to almost any sensor and any model of noise. We justify it as follows. In the real world, the sensor values are subject to a host of unmodeled effects. Imagine we now choose to model all these effects. For example, in our range-finder example, we now choose to model the color of the surface at y, the material, the temperature of the air, and so forth. All of these parameters form a state space which we may term U. To model them, we extend our state space from $C \times M$ to $C \times M \times U$—to include all the error parameters. Hence, a geometric point $y \in M$ together with the "physical" parameter $\epsilon \in U$—which encodes the physical characteristics of the world that cause variation in the sensor—together, y and ϵ determine the effect of the world at point y on the sensor value. This simply says that we regard ϵ as part of the state of the world. That is, we construct the sensing map $\underline{\mathbf{p}}$ to answer question (Q1) (sec. A), with M' substituted for M. By viewing U as "hidden state," we can hence hide a non-deterministic sensing model inside a deterministic sensor map.

Now of course, in general, this kind of modeling of "physical" error parameters will not be possible—we do not, in general, have adequate knowledge of the error characteristics to construct an accurate model of $\underline{\mathbf{p}}$ and U. But, in these cases, we *can* make a worst-case assumption about the error. We note that for a purely non-deterministic, adversarial error model (see [E90] for a characterization of different error types, including non-deterministic, adversarial, probabilistic, etc), it suffices to have the dimension of U and B be the same. We call U the *hidden state* of the world. We assume that the error parameter $\epsilon \in U$ varies

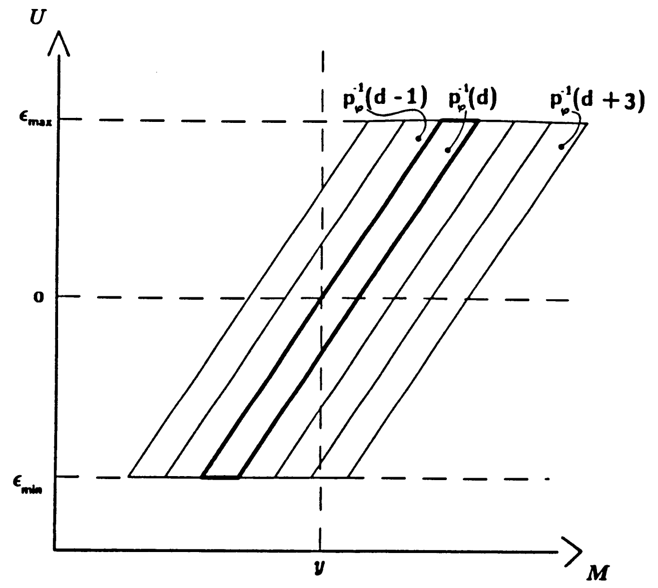

Figure 12: Uncertainty, U, is graphed against the world, M, and the ϵ-perceptual equivalence classes for a simple range-finder are shown.

---

in some uncontrolled manner, and perturbs the sensor readings we obtain. We now see the following:

Claim B.3 *Sensor noise that can be modeled by a non-deterministic sensor map $p^* : C \times M \to B$ with geometric error bounds, can be converted into a deterministic sensor map* $\mathbf{p}$ *with hidden state U. It suffices to have* $\dim U = \dim B$.

Now let us return to our example of p^* and $\mathbf{p}$ in eqs. (21) and (20).

Definition B.4 *Suppose we curry a sensing map with hidden state* $\mathbf{p}$ *(eq. (21)) with $\phi \in C$, to obtain*

$$\mathbf{p}_\phi : M \times U \to B$$

so that $\mathbf{p}_\phi(y, \epsilon) = \mathbf{p}(\phi, y, \epsilon)$. *Then at the fixed configuration ϕ, the* ϵ-perceptual equivalence classes *are classified by sets of the form*

$$\mathbf{p}_\phi^{-1}(b), \tag{22}$$

for a sensor value b in B. That is, each ϵ-perceptual equivalence class is a cell in the arrangement generated by sets of the form (22).

Once we introduce sensor noise, we obtain an augmented framework in which the sensing maps have hidden state. This hidden state models the non-determinism of sensor noise and error. Essentially the ϵ-perceptual equivalence classes now serve in place of the perceptual equivalence classes we had before. To get some intuition for the structure of an ϵ-perceptual equivalence class, consider our simple example of a range-finder. Here, U would simply be some real interval. Suppose y is a point at distance d from x, as shown in fig. 11. The ϵ-perceptual equivalence class for sensor value $\lfloor d \rfloor$ would contain the point $(y, 0) \in M \times U$. This is simply because $\mathbf{p}_\phi(y,0) = \lfloor d{+}e(0) \rfloor = \lfloor d \rfloor$. But the ϵ-perceptual equivalence class $\mathbf{p}_\phi^{-1}(\lfloor d \rfloor)$ would also contain all points (y', ϵ), such that $\mathbf{p}_\phi(y', \epsilon) = \lfloor d \rfloor$, i.e., $\lfloor \|y' - x\| + \epsilon \rfloor = \lfloor d \rfloor$.

Now, in our new model, the world is $M \times U$, i.e., it contains the hidden state U. The ϵ-perceptual equivalence class $\mathbf{p}_\phi^{-1}(b)$ consists of all "points" (y, ϵ) in the world that can result in the sensor value b. For example, see fig. 12. In this case, the ϵ-perceptual equivalence class $\mathbf{p}_\phi^{-1}(b)$ (for an integer b) is a diagonal strip in $M \times U$. The width of the strip is the sensor resolution.

On the other hand, given a point y in M (or more generally, a feature at y), we may wish to ask, *What sensor values can y generate?* The answer is as follows; see fig. 12. We define the *perceptual alias set of y* to be all ϵ-perceptual equivalence classes that "project to cover y" in M. These are the (y', ϵ)-"points" in the world that are "confusable" with y. We define the *semblance* of a point y in M to be all the ways that y can appear as projected through the sensor, when noise is taken into account. More formally: we begin with the "local" definitions (for a fixed configuration in C). By a slight abuse of notation, we overload the projection operator π_M to denote projection from any product of M onto M: below, $\pi_M : M \times U \to M$.

Definition B.5 *For a sensing function* $\mathbf{p}$, *a configuration* ϕ *in* C, *and a point* y *in* M, *the* semblance of y *is the set of the sensor values whose ϵ-perceptual equivalence classes project to cover y. The semblance of y is denoted* $\mathrm{sem}(y)$, *and*

$$\mathrm{sem}(y) = \{\, b \in B \;\big|\; \mathbf{p}_\phi^{-1}(b) \cap \pi_M^{-1}(y) \neq \emptyset \,\}.$$

Next, we can give a rigorous characterization of perceptual aliasing:

Definition B.6 *For a sensing function* $\mathbf{p}$, *a configuration* ϕ *in* C, *and a point* y *in* M, *the* perceptual alias set of y *consists of all ϵ-perceptual equivalence classes whose projection onto M covers y. That is:*

$$\{\, \mathbf{p}_\phi^{-1}(b) \;\big|\; b \in \mathrm{sem}(y) \,\}.$$

Hence, the perceptual alias set of y consists of the ϵ-perceptual equivalence classes of the semblance of y.

The characterization of ϵ-perceptual equivalence classes in fig. 12 is in fact representative for higher dimensions as well. Thus, the ϵ-perceptual equivalence class are "well-behaved." More specifically, It is simple to show:

Claim B.7 *Suppose the error function e is injective. Then for two sensor values b_1 and b_2 in B, $\mathbf{p}_\phi^{-1}(b_1) \cap \mathbf{p}_\phi^{-1}(b_2) \neq \emptyset$ if, and only if, $b_1 = b_2$.* □

Corollary B.8 *The ϵ-perceptual equivalence classes are disjoint if the error function is injective.*

To give some intuition about the structure of the ϵ-perceptual equivalence classes, consider the following. Imagine an LMT-style position sensor, with error. We convert this sensor to a deterministic sensor (such as $\underline{\mathbf{p}}$) with hidden state; in this case, U will be a ball (representing an error ball). The error value $\epsilon \in U$ is used to perturb the position $\phi \in C$ to obtain the sensor value $\phi + e(\epsilon)$. Let us assume for a moment that $M = C = B = \Re^n$, so of course, U is the n-dimensional ball, and the error function is injective. Then, the ϵ-perceptual equivalence classes are disjoint, each ϵ-perceptual equivalence class is an n-dimensional sheet in $M \times U$. (If e is linear, the sheets are planes, etc).

Semblences and perceptual aliases may be globalized in a straightforward manner to a world sensing bundle, $\mathbf{p}$, with domain $C \times M \times U$. See [DJ2] for details.

B.3 Discussion on Noise, Error, and Perceptual Aliasing

What we have essentially done in our construction is graph the state space (M, above) against the error U. We then proceeded to reason about the resulting enlarged state space $M \times U$. It is not surprising that perceptual equivalence classes formed in the product state space were the natural generalizations of perceptual equivalence to the case with noise. Our construction, of course is not new. In particular, we recall the work by Brooks [Brooks 82], in which the uncertainty is graphed against state. Recall also the work by Donald [Don, D, Don90] on compliant motion planning in the presence of model error. In this work, we introduced an "error" space like U and formed a "generalized configuration space" which was the product of the state space and the error space. Many others have used this construction.

Here are some significant aspects of our use of the construction:

1. In most previous work, the uncertainty space U is known ahead of time. In our case it is completely unknown.

2. Suppose we let U represent the actual unknown parameters in the world that corrupt and influence our sensor values (see sec. B.2). We will call this the "concrete" error space in that parameters in this space will determine adumbrations of the sensors' interaction with the physical world, and thereby the sensor values that result from interrogating features in that world. So, for a sonar sensor, U might include temperature, surface material, windspeed, etc. As another example, consider a flux-gate compass. Suppose the robot trusts its odometry,[20] and detects the presence of some magnetic disturbance in the room. But, the disturbance might vary in intensity and direction as the robot moves. In this case U could encode the flux of an unknown, interfering magnetic field. This field would vary spatially. As the robot moved, it could, perhaps, deduce the characteristics of the interfering field–that is, the structure of U. For example, the field might be stronger near the Lisp-Machine in the corner of the lab. By discovering the correlation in field intensity change with spatial locality through experiments (varying its controls) the robot could determine the sensor flow[21] on $C \times U$. At any rate, in the same way that the robots in sec. 8 could reconstruct

[20]Of course, this might just be one of a number of multiple hypotheses. For more on support of multiple hypotheses, see sec. 8.

[21]*Sensor flow* is defined to be the integral of the recognizability gradient. See sec. A.2.

to infer the structure of $C \times M$, a robot could B-encode the RR-graph of the world sensing bundle, and then use or hypothesize local sections to reconstruct the structure of U or the structure of $C \times U$.

3. The "concrete" physical error space may be hard to model or estimate. But for the purposes of this framework, we can approximate it by using an error space that merely has the same dimension as B. This models a pure, non-deterministic sensor noise, with geometric bounds. We call it the "non-deterministic error space U."

   Note however, that the "concrete" error space will typically have a much higher dimension that the non-deterministic error space. However, the non-deterministic error space is always a projection of the concrete error space. By this, we mean that every point in the concrete error space determines a point in the non-deterministic error space, and that it is possible to calculate the sensor value (with noise) using only the latter.

   The problem of reconstructing the concrete error space from the non-deterministic error space is very similar to the problem of reconstructing $C \times M$ from values in B.

For situations where the intransitivity of perceptual equivalence without U makes working in C too difficult, the problem can always be lifted to the enlarged state space $C \times U$, as described in previous sections. However there exists a cost to this solution—the higher dimensional state space—and so, where possible, we prefer to work in C. The research directions highlighted in this section deserve further study.

C Literature Review

Many of the issues in this paper have been encountered or have precursors in earlier work. The constructive theory of recognizablity—a calculus of the observable—and the idea of directing experiments based on information needs—these ideas are important themes in the classical physical sciences and in robotics. Our enterprise is not new! For example, in the context of human experimenters in quantum mechanics, see (eg) R. Sorenson on Ernst Mach's *Contribution to the Analysis of Sensations.* Sorenson concludes that Mach believed: "... there is no such thing as a purely physical event or connection, but only physical events and connections filtered through our senses (our skin)." However, in our problem, the entities performing and analyzing the observations are machines. In robotics, strategies for coping with uncertainty, and reasoning about reachable and recognizable sets has been a theme in robotics research for many years. See, for example: [LMT, Erd, E89, Don, Brost, Can, Bri, DW]. Early work on planning in the presence of uncertainty investigated using skeleton strategies. [Lozano-Pérez 76] proposed a task-level planner called LAMA which used geometric simulation to predict the outcomes of plans, and is one of the earliest systems to address EDR planning. [Taylor] used symbolic reasoning to restrict the values of variables in skeleton plans to guarantee success. [Brooks 82] later extended this technique using a symbolic algebra system. [Dufay and Latombe] implemented a system which addresses learning in the domain of robot motion planning with uncertainty.

[LMT] proposed a formal framework for automatically synthesizing fine-motion strategies in the presence of sensing and control uncertainty. Their method is called the *preimage* framework. [Mason, 83] further developed the preimage termination predicates, addressing completeness and correctness of the resulting plans. [Erd] continued work on the preimage framework, and demonstrated how to separate the problem into questions of *reachability* and *recognizability*. He also showed how to compute preimages using backprojections, which address reachability alone, and designed and implemented the first algorithms for computing backprojections. [Erdmann and Mason] developed a planner which could perform sensorless manipulation of polygonal objects in a tray. Their planner makes extensive use of a representation of friction in configuration space [Erd]. [Buckley] implemented a multi-step planner for planning compliant motions with uncertainty in 3D without rotations. He also developed a variety of new theoretical tools, including a combined spring-damper dynamic model, 3D backprojection and forward projection algorithms, and a finitization technique which makes searching the space of commanded motions more tractable. Donald [Don89,Don90] proposes error detection and recovery (EDR) strategies for planning and executing robot strategies in the presence of uncertainty. Canny [Can] showed the following: when the geometric world models and also the solution trajectories to the differential inclusion govering the control system are semi-algebraic, then general LMT motion strategies can be computed using the theory of real closed fields. These assumptions cover (for example) motion planning with uncertainty under pure translation. Probabilistic planning and execution strategies are developed by [GM89, E89], and greatly enhance the LMT-family framework. [Lumelsky] considers navigation problems for 2DOF robots with perfect dead reckoning but no model of the world. Other planners, and techniques for coping with uncertainty are discussed by [Brost; Caine; Cutkosky; LLS; Laugier; PTP; Briggs; FHS].

Notions of recognizability developed in the LMT family tree are important to us even without world models and the ability to forward- and back-simulate. Erdmann formalized the "effective set of sensor interpretations" using history and sensing [Erd]. Canny called these the *recognizable sets*, and demonstrated a semi-algebraic parameterization for them [Can]. This allowed him to apply very general tools from computational algebra in order to obtain planning algorithms when one has geometric world models and algebraic forward projections. Canny's planner is based on the recognizable sets. Erdmann also made extensive use of these "sensor interpretations sets" in his work on probabilistic strategies [E89].

In this paper, we develop a family of concepts useful in mobile robotics. These include (for example): *perceptual equivalence classes,* the *lattice of recognizable sets*, the *reachability graph of recognizable sets*, the *reconstruction* problem, and a "measure" of the "reactivity" of a system. These ideas are connected with previous work. Reactive systems, behavior-based robots, and the subsumption architecture are discussed by Brooks [Brooks 1985], Mataric [Mat], and Mahadevan and Connell [MC]. The last describe the use of reinforcement learning in the mobile robot domain. In work on learning manipulation strategies, Christiansen, Mason, and Mitchell [CMM] identify perceptual equivalence classes as consequences of imperfect sensing, which may be viewed as a projection [C]. Whitehead and Ballard [WB] and also Shafer [S] discuss perceptual and model aliasing. Shafer [S] describes how model aliasing can partition the set of world states into equivalence classes with the same representation. We believe that [MC, CMM, C, WB, S, Erd, Can] understand and exploit notions of perceptual equivalence classes in a fundamental way. Hager and Mintz [HM] demonstrate methods for sensor fusion and sensor planning based on probabilistic models of uncertainty. Rosenschein [R], in work parallel to ours, considers a logic framework in which one can synthesize automata that build data structures from sensor data that "correlate" accurately with the external world. Such data structures are related to our RR-graphs. Reachability graphs have many ancestors; see even [Min]. Our lattice is reminiscent of Mitchell's *version space* [Mit]. Leonard, Cox, and Durrant-Whyte [LDWC, CL] use notions of observation and expectation that we find significant and these notions anticipate some of our constructions. In [Don90] we suggested the following approach, called *"weak error detection and recovery (EDR)."* Suppose we are given a collection of subgoals that are closed under intersection. Then by considering all unions of subcollections of these subgoals we can obtain a measure of weakest recognizability. This idea anticipates the lattice of recognizable sets, whose roots also reach to Erdmann's work on preimages [Erd]. Finally, we believe the ideas of Rosenschein [R] to be of considerable relevance.See also [Su] for work onintegrated architectures.

For reasons of algorithmic tractability, much of our previous work has concentrated on planning (and execution) that emphasized *reachability* (the places a robot can reach during an action) while using very simple models of *recognizability* (the places a robot can distinguish using sensors). Erdmann first introduced this mathematical distinction in his thesis. Indeed, some foundational work has indeed assumed that recognizability (eg, recognizing attainment of a goal during a continuous control regime) was given by some "oracle" and that only reachability need be considered [Don90,Erd89,Bri,FHS]. This effectively reduces planning to a backwards simulation (under control uncertainty), which is an easier (although important) problem.

However, due to the sensitivity of simulation to modeling errors, this focus on reachability tightly coupled the planning process to relatively accurate models of the environment. To consider sparse or incomplete models of the environment, or mobile robots operating in very uncertain and dynamic domains, we take a different tack. In very uncertain domains, reachability computations (i.e., forward- and back-simulations) are essentially valid only locally, since they can only rely on local models of the world. We are trying to develop a framework for sensing and action in which recognizability is the "senior" partner. To say this precisely: in previous work, we attempted in effect to "reduce" recognizability computations to reachability computations. Hence, the fundamental geometric building blocks for this theory were the reachable sets (from some initial conditions, under particular controls). Since these sets had considerable algebraic and geometric structure, geometric algorithms could be obtained to generate control and sensing strategies—plans, if you will.

In this paper we described a theory of planning, sensing and action in which the fundamental building blocks are, in effect, the "recognizable sets"—that is, the places in the world that the robot can recognize and distinguish between. To this end we observed that viewing the world through sensors partitions the world (locally) into "perceptual equivalence classes."[22] The more information the sensors provide, the "finer" this partition is. Various possible partitions of the world fit into a lattice structure (where one partition is "higher" than another if it is finer). This lattice structure captures the information or knowledge state about the world; the lattice is related to the "version space" of Mitchell [Mit]. Rec-

[22]This term was suggested to us by Dan Huttenlocher.

ognizability is a key theme in the preimage framework of [LMT]; see [LMT,Erd,Don,Can]. Canny [Can] described a planner for LMT strategies based on the recognizable sets. The recognizable sets are related to the "preimages" of [LMT], the "effective sensor interpretation sets" of Erdmann [Erd], the "recognizable sets"[23] of Canny [Can], and the "signature neighborhoods" of Mahadevan and Connell [MC]. They are called "sensor equivalence classes" by Christiansen, Mason, and Mitchell [CMM], "perceptual aliases" by Whitehead and Ballard [WB], and "model aliases" by Shafer [S].

References

[AC] **Aubin, J.P. and Cellina, A.** *Differential Inclusions,* Springer-Verlag: New York (1984).

[BBD+] **Baillieul, Brockett, Donald,** ET AL. *Robotics,* Symposia in Applied Mathematics, vol 41. American Mathematical Society Press, Providence (1990).

[BaKu] **Barshan, B. and R. Kuc** *Differentiating Sonar Reflections from Corners and Planes by Employing an Intelligent Sensor,* IEEE PAMI 12 (**6**), pp 560-569 (1990).

[BSD] **Blondell, B., Shollenberger, S. and Donald, B.** *Design of a microprocessor driven generic controller for a mobile robot base,* Cornell CS Dept., Ithaca, NY CU-CS-TR-90-1140 (1990).

[Bri] **Briggs, Amy.** (1989) *An Efficient Algorithm for One-Step Compliant Motion Planning with Uncertainty,* 5^{th} ACM Symp. Comp. Geom., Saarbrucken, BRD

Brooks, R. A. *Symbolic Error Analysis and Robot Planning,* IJRR Vol 1, no. 4, Dec., 29–68 (1982).

Brooks, R. A. *A Robust Layered Control System for a Mobile Robot,* IEEE JRA **RA-2** (1): 14–23. (1986).

Brost, R., "Automatic Grasp Planning in the Presence of Uncertainty", *IEEE ICRA*, San Francisco, 1986.

[BD] **Brown, R. G. and Donald, B.** *A System for Robotic Navigation, Task-Directed Sensing, and Active Segmentation,* Submitted to *IEEE ICRA*, Nice, 1992 (1991).

[Buc] **Buckley, S. J.** *Planning and Teaching Compliant Motion Strategies,* Ph.D. Thesis. MIT, Dept EECS, 1987. Also MIT-AI-TR-936 (1987).

Caine, M., "Chamferless Assembly of Rectangular Parts in Two and Three Dimensions", S.M. thesis, MIT Dept. of Mech. Eng. 1985.

[Can] **Canny, J.F.** *On The Computability of Fine-Motion Plans,* Proc. IEEE ICRA Scottsdale, AZ (1989).

[23] We are grateful to J. Canny for pointing out the similarity.

[C] **Christiansen, A.** *Manipulation Planning from Empirical Backprojections,* CMU Dept. C.S. (1991).

[CMM] **Christiansen, A., Mason, M., and Mitchell, T.** *Learning Reliable Manipulation Strategies without Initial Physical Models,* IEEE ICRA (1990).

[CL] **Cox, I., and Leonard, J.** *Temporal Integration of Multiple Sensor Observations for Dynamic World Modeling: A Multiple Hypothesis Approach,* NEC Research Institute; Princeton, NJ (1990).

[CR] **Canny, J., and J. Reif,** "New Lower Bound Techniques for Robot Motion Planning Problems", *FOCS* (1987).

[D] **Donald, B. R.** *A Geometric Approach to Error Detection and Recovery for Robot Motion Planning with Uncertainty,* Artificial Intelligence, **37** (1-3), Dec. pp. 223-271. (1988).

[Don90] **Donald, B. R.**, "Planning Multi-Step Error Detection and Recovery Strategies", *IJRR*, **9** (1), Feb. 1990. pp. 3-60.

[Don] **Donald, B. R.** *Error Detection and Recovery in Robotics,* Springer-Verlag (New York): Lecture Notes in CS, vol. 336 (1989).

[DJ] **Donald, B. and Jennings, J.** *Sensor Interpretation and Task-Directed Planning Using Perceptual Equivalence Classes,* IEEE ICRA, Sacramento, CA (1991).

[DJ2] **Donald, B. and Jennings, J.** *Sensor Interpretation and Task-Directed Planning Using Perceptual Equivalence Classes,* Cornell C.S. Dept TR (1991).

[DJ3] **Donald, B. and Jennings, J.** *Perceptual Limits, Perceptual Equivalence Classes, and a Robot's Sensori-Computational Capabilities,* IEEE Intelligent Robots and Systems, Osaka, Japan (1991).

[DJ4] **Donald, B. and Jennings, J.** *Constructive Recognizability for Task-Directed Robot Programming (Extended Abstract),* Submitted to *IEEE ICRA*, Nice, 1992 (1991).

[DW] **Durrant-Whyte, H.** *Concerning Uncertain Geometry in Robotics,* Artificial Intelligence, **37** (1-3), Dec. (1986).

[Erd] **Erdmann, M.** *Using Backprojections for Fine Motion Planning with Uncertainty,* IJRR Vol. 5 no. 1 (1986).

[E89] **Erdmann, M.** *Probabilistic Strategies for Robot Tasks,* MIT AI Lab PhD Thesis, Dept. EECS (1989).

[EM] **Erdmann, M., and M. Mason,** "An Exploration of Sensorless Manipulation", *IEEE International Conference on Robotics and Automation*, San Francisco, April, 1986.

[EMV] **Erdmann, M., Mason, M, and Vanacek G.** *Mechanical Parts Orienting: The Case of a Polyhedron on a Table,* IEEE ICRA (1991).

[FHS] **Friedman, J., J. Hershberger and J. Snoeyink** *Compliant Motion in a Simple Polygon,* 5^{th} ACM Symp. Comp. Geom., Saarbrucken, BRD. (1989).

[GM89] **Goldberg, K., and Mason, M.** *Bayesian Grasping,* IEEE ICRA, 1990 (1989).

[GME] **Goldberg, K., Mason, M., and Erdmann M.** *Generating Stochastic Plans for a Programmable Parts Feeder,* IEEE ICRA (1991).

Grigoryev D. Y., "Complexity of Deciding Tarski Algebra" Jour. Symbolic Computation, special issue on decision algorithms for the theory of real closed fields, (1987).

[HM] **Hager, G., and Mintz, M.** *Computational Methods for Task-Directed Sensor Data Fusion and Sensor Planning,* IJRR (**10**) 4 pp. 285-313 (1991).

[Hut] **Hutchinson, S.** *Exploiting Visual Constraints in Robot Motion Planning,* IEEE ICRA (1991).

[JDC] **J. Jennings, B. Donald, and D. Campbell** *Towards Experimental Verification of an Automated Compliant Motion Planner based on a Geometric Theory of Error Detection and Recovery,* Proc. IEEE ICRA, Scotsdale, Arizona (1989).

[KTB] **Kriegman, D., E. Triendl, and T. Binford** *Stereo Vision and Navigation in Buildings for Mobile Robots,* IEEE TRA 5 (**6**) pp 792-803 (1989).

[KS] **Kuc, R. and M. Siegel** *Physically Based Simulation Model for Acoustic Sensor Robot Navigation,* IEEE PAMI 9 (**6**) pp. 766-778 (1987).

[LDWC] **Leonard, J., Durrant-Whyte, H., and Cox, I.** *Dynamic Map Building for an Autonomous Mobile Robot,* Preprint: To appear in IEEE/RSJ Intl. Conf on Intelligent Robot Systems (IROS) (1990).

[LoP] **Lozano–Pérez, T.** *Spatial Planning: A Configuration Space Approach,* IEEE Trans. on Computers (C–32):108–120 (1983a).

Lozano-Pérez T., Jones J., Mazer, E., O'Donnell, P., Grimson, E., "Handey: A Robot System that Recognizes, Plans, and Manipulates", Proc. IEEE Int. Conf. on Robotics and Automation, 1987.

[Lat] **Latombe, J.-C.** *Robot Motion Planning,* Kluwer (1990).

[LLS] **Latombe, J.C., A. Lazanas, and S. Shekhar** *Motion Planning with Uncertainty: Practical Computation of Non-Maximal Preimages,* IEEE/RSJ Int'l. Workshop on Intelligent Robots and Systems, Sept. 4-6 (1989).

[LMT] **Lozano-Pérez, T., Mason, M. T., and Taylor, R. H.** *Automatic Synthesis of Fine-Motion Strategies for Robots,* IJRR, Vol 3, no. 1 (1984).

[Maes] **Maes, P.** *Designing Autonomous Agents,* MIT/Elsevier : Amsterdam (1990).

[MC] **Mahadevan, S. and Connell, J.** *Automatic Programming of Behavior-based Robots using Reinforcement Learning.,* IBM T.J.Watson Res. Rept. (1990).

[Ma] **Mason, M.T.** *Compliance and force control for computer controlled manipulators,* IEEE Trans. on Systems, Man and Cybernetics (SMC-11):418-432 (1981).

[Ma2] **Mason, M. T.** *Automatic Planning of Fine Motions: Correctness and Completeness,* 1984 IEEE ICRA, Atlanta Ga. (1984).

[Mat] **Mataric, M.** *A model for distributed mobile robot environment learning and navigation,* S.M. Thesis, MIT AI Lab, Dept. EECS (1990).

[Min] **Minsky, M.** *A framework for representing knowledge,* The Psychology of Computer Vision, Ed. P. Winston, McGraw-Hill, pp 211-77 (1975).

[Mit] **Mitchell, T. M.** *An Approach to Concept Learning,* Stanford Univ., Palo Alto, CA TR STAN-CS-78-711 (1978).

[ME] **Moravec, H., and A. Elfes** *High resolution maps from wide angle sonar,* Proc. IEEE ICRA. pp 116-121 (1985).

[Mor] **Moravec, H.** *Sensor fusion in certainty grids for mobile robots,* in A. Casals, ed., *Sensor Devices and Systems for Robotics.* Springer-Verlag (1989).

[Nat] **Natarajan, B. K.** 1986 (Oct. 27–29, Toronto, Ontario). An Algorithmic Approach to the Automated Design of Parts Orienters. *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pp. 132–142.

[OdD] **O'Donnell, M. and Donald, B.** *On Large Scale Planar Manipulation,* Cornell CS Dept., Ithaca, NY CU-CS-TR 90-1127 (1990).

[RC] **Raibert, M., and J. Craig**, "Hybrid Position/Force Control of Manipulators", *Journal of Dynamic Systems, Measurement, and Control*, No. 102, June, 1981, pp. 126-133.

[RD] **Rees, J. and Donald, B.** *Program Mobile Robots in Scheme,* Submitted to *IEEE ICRA*, Nice, 1992 (1991).

[R] **Rosenschein, S.J.** *Synthesizing Information-Tracking Automata from Environment Descriptions,* Teleos Research TR No. 2 (1989).

Schwartz J. and Yap C. K., "Advances in Robotics," Lawrence Erlbaum associates, Hillside New Jersey, (1986).

[S] **Shafer, S.** *Why Can't We Model the Physical World?,* for the CMU CS Dept. 25^{th} Anniversary (1990).

Simunovic, S. N., "An Information Approach to Parts Mating", Ph.D. thesis, Dept. EE, MIT, 1979.

[Su] **Sutton, R.** *Integrated architectures for learning, planning, and reacting based on approximating dynamic programming,* Proc. 7^{th} Intl. Conf. on Machine Learning (1990).

Tarski A., "A Decision Method for Elementary Algebra and Geometry" Univ. of Calif. Press, Berkeley, (1948), second ed. 1951.

Taylor, R. H. *The Synthesis of Manipulator Control Programs from Task-level Specifications,* Stanford A.I. Lab., AIM-282, July (1976).

Ward, B. and McCalla, G. *Error Detection and Recovery in a Dynamic Planning Environment,* AAAI (1983).

[WB] **Whitehead, S. and Ballard, D.** *Active Perception and Reinforcement Learning,* 7^{th} Int'l. Conf. on Machine Learning (1990).

Yap, C., "Algorithmic Motion Planning", in *Advances in Robotics: Volume 1*, edited by J. Schwartz and C. Yap, Lawrence Erlbaum Associates, 1986.

Acknowledgements

Many key ideas in this paper arose in discussions with Tomás Lozano-Pérez, Mike Erdmann, Matt Mason, and Dan Huttenlocher, and we are very grateful to them for discussions and suggestions, and for their generosity with their time and ideas. We would like to thank Leo Dorst for inviting us to write this paper, and for his comments on early drafts. We would like to thank Peter Kahn for many valuable discussions, and for helping us to domesticate the methematics. Many thanks to Prakash Panagaden and Allen Back for discussions and advice. Thanks to Maja Mataric for discussing her work and robotics in general with us. Thanks also to John Canny, Devika Subramanian, Alan Christiansen, David Chapman, and Phil Agre for insight and comments on this work.