2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Abstractions for computing all robotic sensors that suffice to solve a planning problem

Yulin Zhang and Dylan A. Shell

***Abstract*— Whether a robot can perform some specific task depends on several aspects, including the robot's sensors and the plans it possesses. We are interested in search algorithms that treat plans and sensor designs jointly, yielding solutions—i.e., plan and sensor characterization pairs—if and only if they exist. Such algorithms can help roboticists explore the space of sensors to aid in making design trade-offs. Generalizing prior work where sensors are modeled abstractly as sensor maps on p-graphs, the present paper increases the potential sensors which can be sought significantly. But doing so enlarges a problem currently on the outer limits of being considered tractable. Toward taming this complexity, two contributions are made: (1) we show how to represent the search space for this more general problem and describe data structures that enable whole sets of sensors to be summarized via a single special representative; (2) we give a means by which other structure (either task domain knowledge, sensor technology or fabrication constraints) can be incorporated to reduce the sets to be enumerated. These lead to algorithms that we have implemented and which suffice to solve particular problem instances, albeit only of small scale. Nevertheless, the algorithm aids in helping understand what attributes sensors must possess and what information they must provide in order to ensure a robot can achieve its goals despite non-determinism.**

I. Introduction

We currently approach robot sensors from the perspective of consumers, purchasing whatever seems necessary from a catalogue, then writing program code to make robots useful. This perspective puts practical constraints up front: it is influenced by technologies that are currently available, it limits options to what can be fabricated cheaply and sold profitably. Worse, it relies on roboticists to reason (often only heuristically) about the information needed for a robot to achieve its goals. If there is some notion of task structure, reasoning about it is seldom formalized, and may be tied to assumptions often taken for granted (e.g., for fixed price, greater sensor precision is better). This paper approaches the question of sensors from a more fundamental perspective—asking how we might represent and explore *conceivable* sensors. It is, therefore, of a more theoretical nature.

Which sensors are necessary depends on what your robot wants to do. We study robots that act to attain goals while managing uncertainty, formulating these precisely as planning problems, under worst-case non-determinism. Unlike many papers entirely focused on finding plans, this paper examines ways in which sensors affect whether a planning problem can be solved. The perspective is that sensor choices alter the set of feasible plans, and we look at sensor/plan pairs jointly. We examine the space of sensors that are useful with respect to a specific given problem. These sensors, indeed especially those that provide little information, can be enlightening. Still, we do require they provide information to make progress toward goals [1], even in the presence of uncertainty. We are interested in exploring all sensors, including even hypothetical ones, for which there exists some goal-achieving plan.

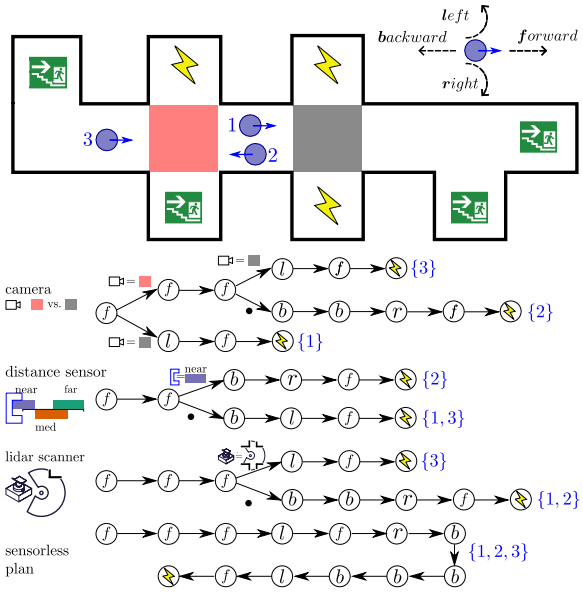

Fig. 1: A wheeled robot (as a blue disk) needs a charging station (the lightning bolts), but is slightly lost (the uncertainty in its initial pose is shown visually, as three possibilities). Unable to navigate stairs, it must avoid those locations lest it topple down a stairwell. The robot is able to recharge its battery despite the presence of uncertainty, with the help of either a camera, a simple linear distance sensor, or a short-range scanning lidar. (If bumping into walls is permitted, a sensorless plan is possible as well.)

Fig. 1 shows a simple didactic scenario illustrating multiple aspects of the problem: a robot, uncertain about its initial position and incapable of navigating stairs, needs to reach a charging station. We give four exemplar sensors that, under different plans, ensure goal attainment:

(i) a camera to distinguish red and gray helps to eliminate uncertainty in the initial pose when following the top plan;

(ii) a robot with a distance sensor can disambiguate initial position 2 from $\{1, 3\}$, since it observes that it is near the wall after two forward moves only when it starts at 2, while observing medium or far from the wall for $\{1, 3\}$;

(iii) with a lidar sensor the robot can distinguish 3 from $\{1, 2\}$ since, after three forward moves from 3, it senses a

Yulin Zhang and Dylan A. Shell are with the Dept. of Computer Science, Texas A&M University, College Station, TX, USA.

This work was supported by the NSF through awards IIS-1453652 and IIS-1527436.

different polygon from those of 2 and 3.

(iv) the vacuous sensor also suffices, albeit only under the assumption of benign collisions, and with many steps.

The sensors do not all quash the uncertainty completely, but they eliminate enough to reach the goal under different plans. For example, the robot with a distance sensor never resolves whether it came from 1 or 3 in executing the corresponding plan. The robot with a lidar sensor does not distinguish 1 from 2. But, in both cases, the robot reaches a charger. There are also important differences in the sensors' fidelity. The camera divides all the locations into three equivalent classes: a red location, a gray one, and the white ones. In contrast, the distance sensor's specification tells us that middle range distance readings are noisy, failing to separate medium and far distances from the wall crisply (when the robot observes 'med', then it is either at a medium or a far range from the wall; when obtaining 'near', it is close to the wall).

Sensors can be modeled as the information they provide for the plan. While previous works [2, 3, 4] regard sensors as partitions over all events to be perceived, this paper is more general, considering sensors as covers. Doing so requires some care, including new representations and means to lessen the combinatorial explosion that a naïve treatment entails.

II. Model

We study a setting depicted in Fig. 2. The robot is equipped with a *sensor*, through which it receives observations from the world. Actions are chosen to alter states according to the robot's *plan* to, ultimately, reach some goal states in the world. The sensor may have limited fidelity and fail to distinguish different observations from the world. The uncertainty in sensing is modeled via a type of function, termed a *sensor map*. These elements are formalized in terms of p-graphs and sensor maps that we outline below.

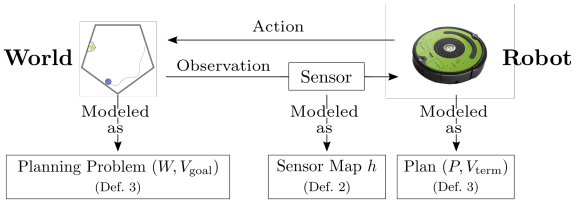

Fig. 2: An overview of the setting: the robot is modeled abstractly as realizing a plan to achieve some goal in the world. The sensor is modeled as a sensor map. Both the world and the plan have concrete representations as p-graphs.

A. A p-graph and its interaction language

We start by defining p-graphs and related properties. For more complete formal definitions, please see [5].

Definition 1 (p-graph [5])**.** A *p-graph* is an edge-labeled directed bipartite graph $G = (V_y \cup V_u, E, Y, U, V_0)$, where

1) the finite vertex set $V(G)$, whose elements are also called *states*, form two disjoint subsets: the *observation vertices* V_y and the *action vertices* V_u, with $V(G) = V_y \cup V_u$;
2) each edge $e \in E$ originating at an observation vertex bears a set of observations $Y(e) \subseteq Y$, containing *observation labels*, and leads to an action vertex;
3) each edge $e \in E$ originating at an action vertex bears a set of actions $U(e) \subseteq U$, containing *action labels*, and leads to an observation vertex; and
4) a non-empty set of states $V_0(G)$ are designated as *initial states*, which may be either action states ($V_0(G) \subseteq V_u$) or observation states ($V_0(G) \subseteq V_y$), exclusively.

We use the word 'event' to mean either an action or an observation. Here, Y is assumed to be finite. A sequence of alternating actions and observations, also called an *execution*, can be traced in the p-graph, if there exists a path, starting from some initial state, with the same number of edges and each edge in the path bears the corresponding event. A p-graph G describes a set of such event sequences.

One p-graph is used to model the world, another the robot. Both p-graphs are coupled, resulting in a planning problem. Sensors influence this coupling relationship by influencing the distinguishability of observations made by the robot. Conflations and corruptions of events are treated next.

B. Sensor maps

Definition 2 (observation/sensor maps [6])**.** A *sensor map* on p-graph G is a function $h : Y \to \mathcal{P}(X) \setminus \{\varnothing\}$ mapping from an observation in Y to a non-empty set of observations X, where $\mathcal{P}(X)$ is the powerset of X.

If h maps y_1 to $\{x_1, x_2, x_3, x_4\}$ then, when event y_1 happens in the world, the robot may receive any of those four values as a sensor reading; further, we assume the choice happens non-deterministically.

Given any subset of sensor readings $X' \subseteq X$ as input to (that is, observed or perceived by) the robot, the associated observations within the world W are related via the preimages of X' under h, denoted by $h^{-1}(X') := \{\ell \in Y(W) \mid h(\ell) \cap X' \neq \varnothing\}$. Below, the notation for a sensor map h and its preimage h^{-1} is extended in the usual manner to p-graphs by applying the function to labels on each observation edge, i.e., in the obvious way.

C. Planning problems and plans

Definition 3 (planning problem and plan)**.** A *planning problem* is a p-graph W along with a goal region $V_{\text{goal}} \subseteq V(W)$; a *plan* is a p-graph P equipped with a termination region $V_{\text{term}} \subseteq V(P)$.

Definition 4 (solves under sensor map)**.** A plan (P, V_{term}) *solves a planning problem* (W, V_{goal}) *under sensor map* h if there is some integer which bounds the length of all joint-executions of W and $h^{-1}\langle P\rangle$, and for each joint-execution and any pair of nodes $(v \in V(h^{-1}\langle P\rangle), w \in V(W))$ reached by that execution simultaneously, the following holds:

1) if v and w are both action nodes and, for every label borne by each edge originating at v, there exist edges originating at w bearing the same action label;
2) if v and w are both observation nodes and, for every label borne by each edge originating at w, there exist edges originating at v bearing the same observation label;
3) if $v \in V_{\text{term}}$ and then $w \in V_{\text{goal}}$;

4) if $v \notin V_{\text{term}}$ then some extended joint-execution exists, continuing from v and w, reaching the termination region.

Property 1) and 2) define a notion of safety; 3) of correctness; 4) of liveness. Note that the sensor map, modeling robot's sensor in this paper, may affect the solvability of the planning problem. In other words, we have to examine the safety when searching for sensor maps.

D. Sensor design in a planning problem

Now we can define the central problem of the paper:

Problem: Joint-Plan-Sensor-Design (JPSD)
Input: A planning problem (W, V_{goal})
Output: All the sensor maps $\mathcal{H}$, such that there exists a plan (P, V_{term}) to solve the planning problem (W, V_{goal}) under each sensor map $h \in \mathcal{H}$.

III. Computational abstractions for sensor maps

Sensor maps map observations to their images, while the planning problem is defined in the preimage space. To solve this problem, we will begin by considering an alternate form in the preimage space for the sensor maps.

A. Equivalent representation for sensor maps

Any sensor map has an equivalent cover representation.

Theorem 1. *For planning problem (W, V_{goal}), any sensor map h is equivalent to a cover up to plan solvability.*

Proof. $\Rightarrow$: Given any sensor map h, to see whether a plan is a solution (cf. Def. 4), we must determine the preimage $h^{-1}(x) = \{\ell \in Y(W) \mid h(\ell) = x\}$ for single readings x. Collect all the data associated with h, on the X, via

$$M = \{h^{-1}(x_1), h^{-1}(x_2), \ldots, h^{-1}(x_n)\},$$

where $X = \{x_1, x_2, \ldots, x_n\}$. This is a multiset. But now observe that where for any x_i and x_j we have $h^{-1}(x_i) = h^{-1}(x_j)$, we can construct a new sensor map by replacing x_i and x_j with a new symbol x'. This new sensor map is also a solution if and only if h is a solution for JPSD. Under this new sensor map, no two readings in the sensor map share the same preimage, and h^{-1} can be thus represented as set

$$C = \{h^{-1}(x_1), h^{-1}(x_2), \ldots, h^{-1}(x_n)\},$$

where $\cup_{x_i \in X} h^{-1}(x_i) = Y(W)$. The set above is called a *cover* for set $Y(W)$. Henceforth, we call the cover for sensor map h an *observation cover*, denote it C_h. (It is a subset of the powerset of $Y(W)$, i.e., $C_h \subseteq \mathcal{P}(Y(W)) \setminus \{\varnothing\}$.)

$\Leftarrow$: Having just showed that there exists a cover interpretation for any sensor map h, we now construct a sensor map for any observation cover. Suppose cover $\{S_1, S_2, \ldots, S_k\} \subseteq \mathcal{P}(Y(W))$ for set $Y(W)$ is given. Taking the first k natural numbers for X, consider a label map h defined so that $y \overset{h}{\mapsto} \{i \in \{1, 2, \ldots, k\} \mid y \in S_i\}$.

Together, the cover C_h is an equivalent representation for any sensor map h, up to plan solvability. □

B. Operations on observation covers

Next, we give two operations on covers (projection and intersection) that are useful for sensor maps.

The sensor map is a cover for all observations in the planning problem. Only some small number of observations may be applicable while at particular world states. We are interested in how the observations in such a reduced set conflate with each other. This is realized via an operation that reduces the domain:

Definition 5 (cover projection)**.** For cover $C = \{G_1, G_2, \ldots, G_n\}$, denote its domain by $d(C) = \cup_{1 \le i \le n} G_i$. Then the *projection of* C on any domain D is $\pi_D(C) = \{G_i \cap D | G_i \in C\}$.

We call sensor map $\pi_D(C)$ with reduced domain $d(C) \cap D$ a *partial sensor map*. The word 'partial' is apt as the sensor map need not cover every observation in the planning problem.

On the other hand, we are also interested in finding all sensor maps with certain behavior on their restrictions. Specifically, we desire to find all label maps which, when given two partial label maps, agree with those label maps on their projections. This comes from an intersection between two partial sensor maps.

Definition 6 (cover intersection)**.** For any two partial sensor maps, expressed as cover C_1 and C_2, with the union of their domains $D = d(C_1) \cup d(C_2)$, then let $\mathbb{D}$ be all covers[1] whose domain is D. Then the *intersection* of C_1 and C_2, denoted $C_1 \sqcap C_2$, is defined so that $\forall C' \in \mathbb{D}$, we have $C' \in C_1 \sqcap C_2$, if and only if

(a) $d(C') = d(C_1) \cup d(C_2)$, and
(b) $\pi_{d(C_1)}(C') \subseteq C_1$ and $\pi_{d(C_2)}(C') \subseteq C_2$.

Note that $\sqcap$ is associative and that $C_1 \sqcap \varnothing = \varnothing$ for any cover C_1. When no cover that satisfies *(a)* and *(b)* above, then $C_1 \sqcap C_2 = \varnothing$. We say that C_1 is *compatible* with C_2 if $C_1 \sqcap C_2 \neq \varnothing$. We will also lift this notation to the intersection of lists of covers. In writing $\mathbb{L}_1 \sqcap \mathbb{L}_2$ for two lists of covers $\mathbb{L}_1$ and $\mathbb{L}_2$, we mean $\mathbb{L}_1 \sqcap \mathbb{L}_2 = \cup_{C_1 \in \mathbb{L}_1, C_2 \in \mathbb{L}_2} C_1 \sqcap C_2$.

IV. Jointly searching for sensor designs & plans

First, we construct a robot's belief tree and then give approaches to search for all sensor designs and plans in it.

A. The belief tree under different sensor maps and actions

The robot's plan must manage uncertainties owing to initial ignorance, action non-determinism, and sensor imperfection. The robot's belief expresses this uncertainty, which we represent as a set of states. Without this, the robot may violate plan safety by trying to execute some action that is not possible in its actual state. The dynamics of the belief will be captured by a finite tree structure, where each vertex lists a set of world states, the robots' belief. Plans need only visit each belief vertex at most once.

[1]Throughout, variables in blackboard bold represent a list of covers.

Theorem 2. *Let $\mathcal{W}$ be the set of estimated world states for the robot's belief. For any sensor design $h \in \mathcal{H}$, where $\mathcal{H}$ is a set of sensor maps for* JPSD*, if there exists a plan that solves the planning problem under h, then there exists another plan, also a solution, that visits $\mathcal{W}$ at most once under h.*

Proof sketch. Construct a plan by shortcutting directly to the last visit to $\mathcal{W}$ under the same sensor map. □

Let $A(w)$ be the set of outgoing events for vertex $w \in V(W)$. Then the belief tree, a sketch of which appears in Fig. 3, can be constructed as follows:

- ▷ *Initialization*: An initial vertex $\mathcal{W}^0$ of the same vertex type is created for the set of initial world states $V_0(W)$.
- ▷ *Expanding action vertex $\mathcal{W}$*: Collect the common actions as $U(\mathcal{W}) = \cap_{w\in\mathcal{W}} A(w)$, i.e., the set of actions each of which is available at every state in $\mathcal{W}$. Now, for any action $a \in U(\mathcal{W})$, consider the transition $\mathcal{W} \xrightarrow{\{a\}} \mathcal{W}'$. If the set of world states $\mathcal{W}'$ has not appeared earlier in the path from $\mathcal{W}^0$ to $\mathcal{W}$, add new belief vertex $\mathcal{W}'$ connected via an edge bearing $\{a\}$. Otherwise, add transition from $\mathcal{W}$ to a vertex $\mathcal{W}_{\text{dummy}}$ to avoid expanding the same belief vertex multiple times.
- ▷ *Expanding observation vertex $\mathcal{W}$*: Let all possible observations at the states in $\mathcal{W}$ be $Y(\mathcal{W})$, i.e., $Y(\mathcal{W}) = \cup_{w\in\mathcal{W}} A(w)$. As before, construct a transition from $\mathcal{W}$ to $\mathcal{W}'$ if $\mathcal{W}'$ is new, or to $\mathcal{W}_{\text{dummy}}$ otherwise. But now do this, not just the singletons, but for every $G \subseteq Y(\mathcal{W})$.
- ▷ *Goals in the tree*: Mark $\mathcal{W}$ a goal state, if $\mathcal{W} \subseteq V_{\text{goal}}$.

The belief tree is finite. Any sensor map and goal-achieving plan are a subtree that satisfies the following:

- (i) *Goals are achieved:* the leaf vertices in the subtree are all in the goal region;
- (ii) *Readiness to receive all observations:* the outgoing labels at a particular observation vertex in the subtree cover all outgoing events in the original belief tree;
- (iii) *Discernment is consistent:* the subset of observations in the tree is universal, i.e., if $\{o_1, o_2\}$ appears on any edge of the subtree, then it will appear at every belief vertex whose outgoing events contains both o_1 and o_2.

B. Searching for sensor designs and plans jointly

Next, we search this structure for sensor designs and plans jointly, returning all appropriate sensor maps. While the tree is constructed from the root down, this search bubbles from the leaves back upwards.

For each belief vertex $\mathcal{W}$, we will maintain a list of covers, denoted by $\mathbb{L}(\mathcal{W})$, to record all the appropriate observation covers in the subtree. When $\mathcal{W}$ is in the goal region, there are no constraints on sensor maps from its subtree. Hence, we create a new symbol $\epsilon \notin Y$, and initialize its cover list to $\mathbb{L}(\mathcal{W}) = [\![\{\epsilon\}]\!]$. This will make it compatible with any cover when integrating with the goal-achieving sensor covers in a bottom-up manner. For any non-goal belief vertex $\mathcal{W}^p$ ('p' stands for parent), we will construct its cover list from its children. Let the outgoing events be $\{G_1, G_2, \ldots, G_m\}$ and the corresponding child vertices be $\{\mathcal{W}^c_1, \mathcal{W}^c_2, \ldots, \mathcal{W}^c_m\}$ ('c' for child). Then we have:

- If $\mathcal{W}^p$ is an action vertex, then each cover in any of its children's lists $\mathbb{L}(\mathcal{W}^c_i)$ is a valid one for $\mathcal{W}^p$ (under a particular action choice), i.e., $\mathbb{L}(\mathcal{W}^p) = \cup_{1\le i\le m}\mathbb{L}(\mathcal{W}^c_i)$.
- If $\mathcal{W}^p$ is an observation vertex, we must consider the combinations from $\{G_1, G_2, \ldots, G_m\}$ that nevertheless cover $Y(\mathcal{W}^p)$. Let K denote one such combination, then $K = \{G_{k_1}, G_{k_2}, \ldots, G_{k_\ell}\}$, where $k_j \in \{1, 2, \ldots, m\}$ and $\cup_{1\le j\le \ell} G_{k_j} = Y(\mathcal{W}^p)$. Each edge labeled with G_{k_j} gives a child vertex $\mathcal{W}^c_{k_j}$, where that child has a cover list $\mathbb{L}(\mathcal{W}^c_{k_j})$ modeling the sensors that can reach the goal from $\mathcal{W}^c_{k_j}$. For a given combination K, representing a set of sensor readings, we want to find all sensor maps, denoted as $\mathbb{L}_K$, that can generate K when projected to $d(K)$, and is goal-achieving for the subtrees starting from each child vertex $\mathcal{W}^c_{k_j}$. This is realized via intersection operations: $\mathbb{L}_K = \cup_{C_1\in\mathbb{L}(\mathcal{W}^c_{k_1}),\ldots,C_n\in\mathbb{L}(\mathcal{W}^c_{k_n})} K \sqcap C_1 \sqcap \ldots C_{m-1} \sqcap C_m$. The $\sqcap$ operation guarantees the universality of the subsets in the resulting cover. Let $\mathbb{C}$ be the set of all such combinations, such that their labels cover $Y(\mathcal{W}^p)$. Each combination $K \in \mathbb{C}$ gives a list of covers for the parent vertex. So we update $\mathbb{L}(\mathcal{W}^p)$ to value $\cup_{K\in\mathbb{C}}\mathbb{L}_K$.

By propagating the list of covers from the goal vertices back upward until the initial belief vertex, we are able to obtain all the covers from $\mathbb{L}(\mathcal{W}^0)$ where there exists some plan for each cover in $\mathbb{L}(\mathcal{W}^0)$ toward the goal.

C. Compact representation with upper covers

In the data structure above, we need to maintain a list of covers $\mathbb{L}(\mathcal{W})$ for each belief vertex $\mathcal{W}$. The list can grow very large. Luckily, we only need to maintain the largest covers among the ones with the same domain. Every subset of such covers is also a valid solution, so long as it is a proper cover.

Theorem 3. *If C is an observation cover in the solution of* JPSD*, then for any $C' \subseteq C$, such that $d(C') = d(C)$, there*

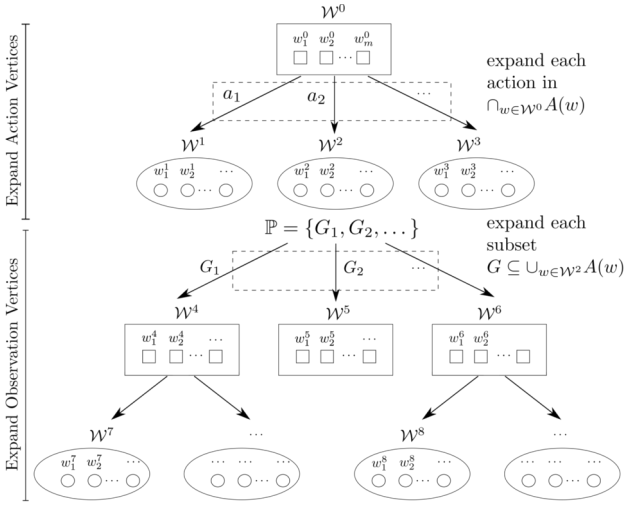

Fig. 3: The robot's belief tree. Action and observation vertices, visualized as boxes and circles respectively, have different expansions.

exists a plan achieving the goal.

This theorem can be proved by showing that the subtree without edges bearing subsets of events in $C' \setminus C$, still has all leaf vertices as goals and sensor map as a valid cover.

Definition 7 (upper cover)**.** Let $\mathbb{C}$ be a list of observation covers, C is an *upper cover* in $\mathbb{C}$ if there does not exist any cover $C' \in \mathbb{C}$, such that $C' \supsetneq C$ and $d(C') = d(C)$.

According to Theorem 3, we only need to maintain a set of upper covers in each $\mathbb{L}(\mathcal{W})$.

D. Empirical explorations of sensor maps

We implemented the algorithms in Python to search for all sensor map solutions for the problem displayed in Fig. 4, a modified version of the (Fig. 1) motivating example: a robot, initially located at 1 or 2, moves to the charging station. The robot can only move forward one or two steps, turn left or right at the location 5 or the corner 6. The robot must avoid bumping into the walls of the four offices A–D and also the stairs. It must, thus, obtain information from its sensors to reduce its uncertainty. To realize this scenario, we construct a world p-graph with 22 states and 11 observations.

The algorithm outputs an upper cover with 767 entries. By enumerating all subsets of the upper cover that covers all the observations (Theorem 3), an enormous number of sensor maps are produced. Among these, several are directly recognizable sensors. For example, they include a sensor map distinguishing every pair of positions, describing a GPS device. The sensor partitioning the situations into those before and after bumping into walls, could be realized as a contact sensor.

Naturally, some of the sensor maps are inscrutable and there are others for which not known hardware implementation could be discerned. For instance, the sensor isolating cell 5 when facing west from cell 7 when facing north (e.g., a distance sensor won't work). This motivates the next section.

V. Structure and Fabrication Constraints for Realizable Sensors

The covers found via the preceding approach might be thought of as a sort of 'free object', on which we may now impose additional constraints. Specifically we're interested in including constraints that help model aspects pertinent to realizable sensors.

A. Sensor map properties

We will start with the following properties:

Property 1 (Partition)**.** Cover $C = \{G_1, G_2, \ldots, G_n\}$ is a *partition*, if $G_i \cap G_j = \varnothing$ for any $i, j \in \{1, \ldots, n\}$, $i \neq j$.

The label map for the camera in Fig. 1 is a partition, as it divides the space into red, gray and white locations.

The next concept of interest is a notion of contiguousness, but we need a more basic structure first.

Definition 8 (Neighbor)**.** Relation $N \subseteq Y \times Y$, written $y_1 N y_2$, is a *neighbor relation* if it is reflexive and commutative.

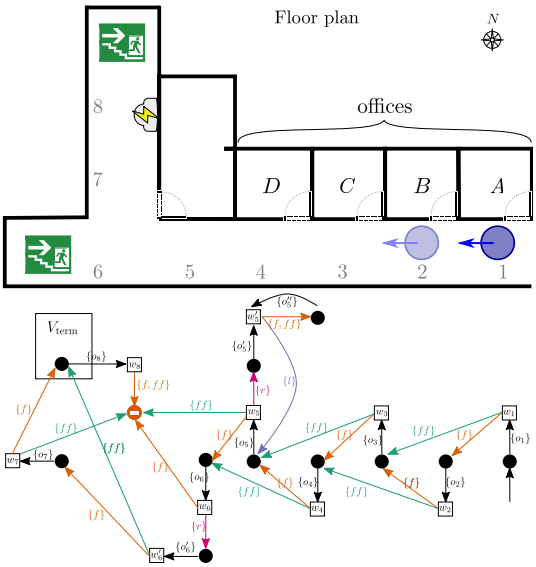

Fig. 4: A robot moves toward the charging station, while avoiding stairs. The figure below shows the p-graph of this planning problem.

Property 2 (Contiguous)**.** With neighbor relation N, then $\mathcal{C}$ is the largest contiguous cover if (1) $\forall y \in Y$, $\{y\} \in \mathcal{C}$; (2) $\forall G_1, G_2 \in \mathcal{C}$, $G_1 \cup G_2 \in \mathcal{C} \iff \exists y_1 \in G_1, \exists y_2 \in G_2$, such that $y_1 N y_2$. A given cover C is *contiguous*, if $C \subseteq \mathcal{C}$.

The distance sensor in Fig. 1 has a contiguous sensor map for the obvious neighbor notion, since its noise distribution is contiguous.

Property 3 (output)**.** Cover C is *k-outputting*, if $|C| = k$.

The cardinality of the sensor keeps track of total number of output readings, a sort of notion of dynamic range.

Property 4 (overlap)**.** A cover $C = \{G_1, G_2, \ldots, G_n\}$ is *k-overlapping*, if $\forall i, j \in \{1, \ldots, n\}$ and $i \neq j$, $|G_i \cap G_j| \leq k$.

This is a generalization of the partition property, quantifying how much readings bleed into one another.

Property 5 (width)**.** Cover $C = \{G_1, G_2, \ldots, G_n\}$ is *k-wide* (or, has width k), if $\forall 1 \leq i \leq n$, $|G_i| = k$.

The width of a cover gives a notion of precision, a sense of the volume of noise, describing the number of events that could account for a single sensor reading.

The properties above may also be combined in specifying constraints on sensor maps.

All of the properties can be used either in (1) reducing the sets generated, or in (2) filtering to discard those which violate the constraints, as operators are applied. For instance, in the first case, if searching for partitions only, then partitions exclusively need be computed—a process easier to write and faster to execute than the full cover case.

B. Empirical search for sensors under fabrication constraints

We included the properties described above in our implementation and examined in the following scenario. A

robot moves along a cyclic track toward some goal, marked by a star. The robot can move forward or backward at different speeds at different parts of the track, which discretizes the track into 6 segments $\{s_1, s_2, \ldots, s_6\}$ as shown in Fig. 5. The angular range of segment s_i is denoted as $\{o_i\}$ for $i \in \{1, \ldots, 4\}$, and $\{o_i, o\}$ for $i \in \{5, 6\}$. The overlap o is the common angular range for both s_5 and s_6, arising from the kink. Now, the set of all observations is $Y = \{o, o_1, o_2, \ldots, o_6\}$, where each observation represents a range of angles.[2] The neighbor relationship of these observations inherits from the circular neighbor relationship of their angles as shown in the figure. Considering only forward or backward actions, the robot, initially located at s_1 or s_3, must move to reach the goal s_5. To achieve this, the robot has to reduce its uncertainty, and it does this via a VHF omnidirectional range (VOR) sensor. As shown on the right-hand side of Fig. 5, the sensor measures the angular information via a timer. The specification of the timer determines the properties of the sensor map. Suppose we have a timer with no noise, then it gives 1-overlapping contiguous sensor maps, such as $[\{o_2, o_3, o_4, o_5\}, \{o_5, o\}, \{o, o_6, o_1\}]$. There are only 2183 such sensor maps. But with a noisy timer, it generates contiguous sensor maps, which leads to 235 807 observation covers.

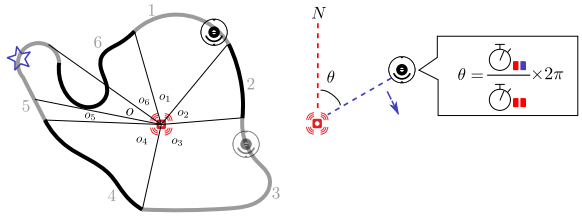

Fig. 5: A robot with a sensor equipped to determine angles moves from its initial position toward the goal along a cyclic track. The sensor is realized by a VOR-like beacon at the center, a photo-electric sensor and a timer on the robot. The beacon has a unidirectional blue light rotating at a fast constant angular velocity, which is so fast that can be neglected with respect to robot's movement. It also emits an omnidirectional red light when the blue light points North. The robot can determine angular information by timing the difference between seeing red-red and red-blue flashes.

Consideration of the scenario above leads to the following:

Proposition 1. A noiseless sensor taking measurements on a continuous or non-continuous space always gives a 1-overlapping sensor map under discretization.

Proof. When there is no noise for the sensor, the sensor map partitions the original continuous or non-continuous measurement space. The task may only need a coarser discretization of the measurement space. If every boundary of the sensor map is a discretization boundary, then the sensor map is still a partition on the discretized space. If it is not, then there exists a sensor map boundary that falls into one of the discretized observations. That observation is shared by the preimage of only the readings separated by the corresponding sensor map boundary. Hence, the maximum overlap between subsets in the observation cover is 1. □

[2]Previously we pointed out that Y was finite; this is still true, though the elements it contains are themselves infinite sets.

VI. Generalization to Belief Stipulations

Some prior work has examined instances wherein a robot should be stopped from knowing too much due to privacy considerations [7, 8]. In these cases, one may pose constraints on robot's belief; in our prior work this was achieved via logical expressions [4]. To search for sensor maps and plans that also satisfy these richer stipulations, the algorithm above needs the following modifications:

- The belief tree should only contain belief vertices satisfying the stipulations, and the dummy vertex. We transition to the dummy if the target belief violates the stipulations.
- We must expand the action vertex in the belief tree over all subsets of actions in the plan instead of just the singleton ones, since none of the singleton actions may transition to the subtree that satisfies the stipulations in each belief vertex.

VII. Related Work

Approaches for automated design of robots have been the subject of three recent workshops at RSS and ICRA over the last 3 years [9]. Current research examines aspects of hardware fabrication (e.g., 3D-printing [10] and prototyping [11, 12]), interconnection optimization [13], rapid end-to-end development and deployment [14, 15], automated synthesis (jointly for mechanisms and controllers) from specifications of desired capabilities [16], and optimization subject to functionality–resource interdependencies [17, 18].

A rich history of robotics research has examined the information required to accomplish a particular task, including specifically what sensors ought to provide [1]. Since sensors can be costly and unreliable, important early papers explored how one might forgo them entirely [19, 20]; other work examined how one might reason about sensors to establish that they do provide enough information [21, 22]. Our prior work [8] exploits properties of carefully conceived sensors. We are interested in all possible sensors, including hypothetical ones, that provide adequate information to solve the planning problem. Imperfection in sensors is modeled as conflation in the perceived events. This conflation is usually considered to be transitive in existing work [22, 4, 23, 2, 24], when reasoning about the information through the sensors. This paper describes sensors via a sensor map representation which can model non-transitive conflation, in the spirit of [6, 5]. It contributes methods to search for all sensors such that there exists a plan for each one to accomplish a given task.

VIII. Conclusion

This paper explores the space of all sensors that provide enough information to solve the planning problem for the robots. The abstraction used for sensors is a generalization of prior models. A notion of upper cover is proposed to compress the representation and speed the search process. Properties are introduced to express domain-knowledge regarding fabrication constraints for sensors.

References

[1] M. A. Erdmann, "Understanding action and sensing by designing action-based sensors," *International Journal of Robotics Research*, vol. 14, no. 5, pp. 483–509, 1995.

[2] S. Ghasemlou, J. M. O'Kane, and D. A. Shell, "Delineating boundaries of feasibility between robot designs," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and System*, Madrid, Spain, 2018, pp. 422–429.

[3] S. Ghasemlou and J. M. O'Kane, "Accelerating the construction of boundaries of feasibility in three classes of robot design problems," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and System*, Macau, China, 2019, (To appear).

[4] Y. Zhang, D. A. Shell, and J. M. O'Kane, "Finding plans subject to stipulations on what information they divulge," in *Proceedings of International Workshop on the Algorithmic Foundations of Robotics*, Mérida, México, 2018.

[5] F. Z. Saberifar, S. Ghasemlou, D. A. Shell, and J. M. O'Kane, "Toward a language-theoretic foundation for planning and filtering," *International Journal of Robotics Research—in WAFR'16 special issue*, vol. 38, no. 2-3, pp. 236–259, 2019.

[6] F. Z. Saberifar, S. Ghasemlou, J. M. O'Kane, and D. A. Shell, "Set-labelled filters and sensor transformations." in *Robotics: Science and Systems*, Ann Arbor, Michigan, 2016.

[7] J. M. O'Kane, "On the value of ignorance: Balancing tracking and privacy using a two-bit sensor," in *Proceedings of International Workshop on the Algorithmic Foundations of Robotics*, Berlin, Heidelberg, 2008, pp. 235–249.

[8] Y. Zhang and D. A. Shell, "Complete characterization of a class of privacy-preserving tracking problems," *International Journal of Robotics Research—in WAFR'16 special issue*, vol. 38, no. 2-3, pp. 299–315, 2019.

[9] A. Q. Nilles, D. A. Shell, and J. M. O'Kane, "Robot Design: Formalisms, Representations, and the Role of the Designer," in *IEEE ICRA Workshop on Autonomous Robot Design*, Brisbane, Australia, May 2018, https://arxiv.org/abs/1806.05157.

[10] S. Fuller, E. Wilhelm, and J. Jacobson, "Ink-jet printed nanoparticle microelectromechanical systems," *Journal of Microelectromechanical Systems*, vol. 11, no. 1, pp. 54–60, 2002.

[11] A. M. Hoover and R. S. Fearing, "Fast scale prototyping for folded millirobots," in *Proceedings of IEEE International Conference on Robotics and Automation*, Pasadena, California, 2008.

[12] I. Fitzner, Y. Sun, V. Sachdeva, and S. Revzen, "Rapidly prototyping robots: Using plates and reinforced flexures," *IEEE Robotics & Automation Magazine*, vol. 24, no. 1, pp. 41–47, 2017.

[13] J. Ziglar, R. Williams, and A. Wicks, "Context-aware system synthesis, task assignment, and routing," *CoRR*, vol. abs/1706.04580, 2017.

[14] K. S. Luck, J. Campbell, M. Jansen, D. Aukes, and H. B. Amor, "From the lab to the desert: Fast prototyping and learning of robot locomotion," in *Robotics: Science and Systems*, Cambridge, Massachusetts, 2017.

[15] A. Schulz, C. Sung, A. Spielberg, W. Zhao, R. Cheng, E. Grinspun, D. Rus, and W. Matusik, "Interactive robogami: An end-to-end system for design of robots with ground locomotion," *International Journal of Robotics Research*, vol. 36, no. 10, pp. 1131–1147, 2017.

[16] A. M. Mehta, J. DelPreto, K. W. Wong, S. Hamill, H. Kress-Gazit, and D. Rus, "Robot creation from functional specifications," in *Springer Proceedings in Advanced Robotics*, 2018, pp. 631–648.

[17] A. Censi, "A Class of Co-Design Problems With Cyclic Constraints and Their Solution," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 96–103, 2017.

[18] A. Pervan and T. D. Murphey, "Low complexity control policy synthesis for embodied computation in synthetic cells," in *Proceedings of International Workshop on the Algorithmic Foundations of Robotics*, Mérida, México, 2018.

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

[20] M. T. Mason, "Kicking the Sensing Habit," *AI Magazine*, vol. 14, no. 1, pp. 58–59, 1993.

[21] B. R. Donald, "On Information Invariants in Robotics," *Artificial Intelligence — Special Volume on Computational Research on Interaction and Agency, Part 1*, vol. 72, no. 1–2, pp. 217–304, 1995.

[22] J. M. O'Kane and S. M. LaValle, "On comparing the power of robots," *International Journal of Robotics Research*, vol. 27, no. 1, pp. 5–23, 2008.

[23] Y. Zhang, D. A. Shell, and J. M. O'Kane, "What does my knowing your plans tell me?" in *IEEE/RSJ IROS Workshop—Towards Intelligent Social Robots: From Naive Robots to Robot Sapiens*, Madrid, Spain, 2018.

[24] A. Kulkarni, S. Srivastava, and S. Kambhampati, "A unified framework for planning in adversarial and cooperative environments," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, Honolulu, Hawaii, 2019, pp. 2479–2487.