Abstract—The increasing use of smart sensors that can dynamically adapt their observations has created a need for algorithms to control the information acquisition process. While such problems can usually be formulated as stochastic control problems, the resulting optimization problems are complex and difficult to solve in real-time applications. In this paper, we consider sensor management problems for sensors that are trying to find and classify objects. We propose alternative approaches for sensor management based on receding horizon control using a stochastic control approximation to the sensor management problem. This approximation can be solved using combinations of linear programming and stochastic control techniques for partially observed Markov decision problems in a hierarchical manner. We explore the performance of our proposed receding horizon algorithms in simulations using heterogeneous sensors, and show that their performance is close to that of a theoretical lower bound. Our results also suggest that a modest horizon is sufficient to achieve near-optimal performance.

I. INTRODUCTION

Recent advances in embedded computing have introduced a new generation of sensors that have the capability of adapting their sensing dynamically in response to collected information. For instance, unmanned aerial vehicles (UAVs) have multiple sensors—radar and electro-optical cameras—which can dynamically change their fields of view and measurement modes. These advances have created a need for a commensurate theory of sensor management (SM) and control to ensure that relevant information is collected for the mission of the sensor system given the available sensor resources. There are numerous applications involving surveillance, diagnosis and fault identification that require such control.

One of the earliest examples of SM arose in the context of search, with applications to anti-submarine warfare [1]. Sensors had the ability to move and allocate their search effort over time and space. Most of the early work on search theory focused on open-loop search plans rather than feedback control of search trajectories [2]. Extensions of search theory to problems requiring adaptive feedback strategies have been developed in some restricted contexts [3].

Adaptive SM has its roots in the field of statistics, where Bayesian experiment design was used to configure subsequent experiments based on observed information. Wald [4], [5] considered sequential hypothesis testing with costly observations. Lindley [6] and Kiefer [7] expanded the concepts to include variations in potential measurements. Chernoff [8] and Fedorov [9] used Cramer-Rao bounds for selecting sequences of measurements for nonlinear regression problems. Most of the strategies proposed for Bayesian experiment design involve single-step optimization criteria, resulting in greedy or myopic strategies that optimize bounds on the expected performance after the next experiment. Other approaches to adaptive SM using single-stage optimization have been proposed using alternative information theoretic measures [10], [11].

Feedback control approaches to SM that consider optimization over time have also been explored. Athans [12] considered a two-point boundary value approach to controlling the error covariance in linear estimators by choosing the measurement matrices. Multi-armed bandit formulations have been used to control individual sensors in applications related to target tracking [13], [14]. Such approaches are restricted to single-sensor control, selecting among individual subproblems to measure, in order to obtain solutions using Gittins indices [15], [16]. Approximate dynamic programming (DP) techniques have also been proposed using approximations to the optimal cost-to-go based on information theoretic measures evaluated using Monte Carlo techniques [17], [18]. A good overview of these techniques is available in [19].

The above approaches for dynamic feedback control are limited in application to problems with a small number of sensor-action choices and simple constraints because the algorithms must enumerate and evaluate the various control actions. In [20], combinatorial optimization techniques are integrated into a DP formulation to obtain approximate stochastic dynamic programming (SDP) algorithms that extend to large numbers of sensor actions. Subsequent work in [21] derived an SDP formulation using partially observed Markov decision processes (POMDPs) and obtained a computable lower bound to the achievable performance of feedback strategies for complex multi-sensor management problems. The lower bound was obtained by a convex relaxation of the original combinatorial POMDP using mixed strategies and averaged constraints. However, the results in [21] do not specify algorithms with performance close to the lower bound.

In this paper, we develop and implement algorithms for the efficient computation of adaptive SM strategies for complex problems involving multiple sensors with different observation modes and large numbers of objects. The algorithms are based on using the lower bound formulation from [21] as an objective in a receding horizon (RH) optimization problem and developing techniques for obtaining feasible
decisions from the mixed strategy solutions. The resulting algorithms are scalable to large numbers of tasks, and suitable for real-time SM. We also extend the model of [21] to incorporate search actions in addition to classification. We evaluate alternative approaches for obtaining feasible decision strategies, and evaluate the resulting performance of the RH algorithms using multi-sensor simulations. Our simulation results demonstrate that our RH algorithms achieve performance comparable to the predicted lower bound of [21] and shed insight into the relative value of different strategies for partitioning sensor resources either geographically or by sensor specialization.

The rest of this paper is organized as follows: Section II describes the formulation of the stochastic SM problem. Section III provides an example of the column generation technique for generating mixed strategies for SM. Section IV discusses how we create feasible, sequenced sensor schedules from these mixed strategies. Section V documents our simulation results for various scenarios. Section VI summarizes our results and discusses areas for future work.

II. PROBLEM FORMULATION AND BACKGROUND

The problem formulation is an extension of the POMDP formulation presented in [21]. Assume that there are a finite number of locations 1, ..., N, each of which may have an object with a given type, or which may be empty. Assume that there is a set of S sensors, each of which has multiple sensor modes, and that each sensor can observe one and only one location at each time with a selected mode.

Let \( x_i \in \{0,1,...,D\} \) denote the state of location \( i \), where \( x_i = 0 \) if location \( i \) is unoccupied, and otherwise \( x_i = k > 0 \) indicates location \( i \) has an object of type \( k \). Let \( \pi_i(0) \in \mathbb{R}^{D+1} \) be a discrete probability distribution over the possible states for the \( i^{th} \) location for \( i = 1, \ldots, N \) where \( D \geq 2 \). Assume additionally that the random variables \( x_i, i = 1, \ldots, N \) are mutually independent. This assumption limits learning about the nature of an object through observations of other objects.

There are \( s = 1, \ldots, S \) sensors, each of which has \( m = 1, \ldots, M_s \) possible modes of observation. We assume there is a series of \( T \) discrete decision stages where sensors can select which location to measure, where \( T \) is large enough so that all of the sensors can use their available resources. At each stage, each sensor can choose to employ one and only one of its modes on a single location to collect a noisy measurement concerning the state \( x_i \) at that location. Each sensor \( s \) has a limited set of locations that it can observe, denoted by \( O_s \subseteq \{1, \ldots, N\} \). A sensor action by sensor \( s \) at stage \( t \) is a pair:

\[
u_s(t) = (i_s(t), m_s(t))\] (1)

consisting of a location to observe, \( i_s \in O_s \), and a mode for that observation, \( m_s \).

Sensor measurements are modeled as belonging to a finite set \( y \in \{1, \ldots, L_s\} \). The likelihood of the measured value is assumed to depend on the sensor \( s \), sensor mode \( m \), location \( i \) and on the true state at the location \( x_i \) but not on the states of other locations. Denote this likelihood as \( P(y|x_i,i,s,m) \). We assume that this likelihood is time-invariant, and that the random measurements \( y_{i,s,m}(t) \) are conditionally independent of other measurements \( y_{j,s,n}(\tau) \) given the location states \( x_i,x_j \) for all sensor modes \( m,n \) provided \( i \neq j \) or \( \tau \neq t \).

Each sensor has a limited quantity of \( R_i \) resources available for measurements. Associated with the use of mode \( m \) by sensor \( s \) on location \( i \) is a resource cost \( r_s(u_s(t)) \), representing power or another type of resource required to use this action by sensor \( s \). The sensing actions are constrained as

\[
\sum_{t=0}^{T-1} r_s(u_s(t)) \leq R_s \ \forall s \in [1, \ldots, S] \] (2)

This is a hard constraint for each realization of observations and decisions.

Let \( I(t) \) denote the sequence of past sensing actions and measurement outcomes up to and including stage \( t-1 \):

\[
I(t) = \{(u_s(k),y_s(k)), s = 1, \ldots, S; k = 0, \ldots, t-1\}
\]

Under the assumption of conditional independence of measurements and independence of individual states at each location, the conditional probability of \( (x_1, \ldots, x_N) \) given \( I(t) \) can be factored as a product of belief states at each location. Denote the belief state at location \( i \) as \( \pi_i(t) = p(x_i|I(t)) \). When a sensor measurement is taken, the belief state is updated according to Bayes’ Rule. A measurement of location \( i \) with the sensor-mode combination \( u_s(t) = (i,m) \) at stage \( t \) that generates observable \( y(t) \) updates the belief vector as:

\[
\pi_i(t+1) = \frac{\text{diag}\{P(y(t)|x_i = j,i,s,m)\} \pi_i(t)}{\text{I}^T \text{diag}\{P(y(t)|x_i = j,i,s,m)\} \pi_i(t)} \]

where \( \text{I} \) is the \( D+1 \) dimensional vector of all ones. Eq. (3) captures the relevant information dynamics that SM controls.

In addition to information dynamics, there are resource dynamics that characterize the available resources at stage \( t \). The dynamics for sensor \( s \) are given as:

\[
R_s(t+1) = R_s(t) - r_s(u_s(t)); \quad R_s(0) = R_s \] (4)

These dynamics constrain the admissible decisions by a sensor, in that it can only use modes that do not use more resources than are available.

Given the final information \( I(T) \), the quality of the information collected is measured by making an estimate of the state of each location \( i \) given the available information. Denote these estimates as \( v_i, i = 1, \ldots, N \). The Bayes’ cost of selecting estimate \( v_i \) when the true state is \( x_i \) is denoted as \( c(x_i, v_i) \in \mathbb{R} \) with \( c(x_i, v_i) \geq 0 \). The objective of the SM stochastic control formulation is to minimize:

\[
J = \sum_{i=1}^{N} \mathbb{E}[c(x_i, v_i)] \] (5)

by selecting adaptive sensor control policies and final estimates subject to the dynamics of Eq. (3) and the constraints of Eq. (4) and Eq. (2).
The results in [21] provide an SDP algorithm to solve the above problem, with cost-to-go at stage \( t \) depending on the joint belief state \( \pi(t) = [\pi_1(t), \ldots, \pi_N(t)] \) and the residual resource state \( R(t) = [R_1(t), \ldots, R_S(t)] \). Because of this dependency, the cost-to-go does not decouple over locations. This leads to a very large POMDP problem with combinatorially many actions and an underlying belief state of dimension \((D + 1)^N\) that is computationally intractable unless there are very few locations.

In [21], the above stochastic control problem was replaced with a simpler problem that provided a lower bound on the optimal cost, by expanding the set of admissible strategies, replacing the constraints of Eq. (2) by the “soft” constraints:

\[
E\left[\sum_{t=0}^{T-1} r_s(u_s(t))\right] \leq R_s \quad \forall s \in [1 \ldots S] \tag{6}
\]

To solve the simpler problem, [21] proposed incorporation of the soft constraints in Eq. (6) into the objective function using Lagrange multipliers \( \lambda_s \) for each sensor \( s \). The augmented objective function is:

\[
\tilde{J}_\lambda = J + \sum_{t=0}^{T-1} \sum_{s=1}^{S} \lambda_s E[r_s(u_s(t))] - \sum_{s=1}^{S} \lambda_s R_s \tag{7}
\]

A key result in [21] was that when the optimization of Eq. (7) was done over mixed strategies for given values of Lagrange multipliers \( \lambda_s \), the stochastic control problem decoupled into independent POMDPs for each location, and the optimization could be performed using feedback strategies for each location \( i \) that depended only on the information collected for that location, \( I_i(t) \). These POMDPs have an underlying information state-space of dimension \( D + 1 \), corresponding to the number of possible states at a single location, and can be solved efficiently. Because the measurements and possible sensor actions are finite-valued, the set of possible SM strategies \( \Gamma \) is also finite. Let \( Q(\Gamma) \) denote the set of mixed strategies that assign probability \( q(\gamma) \) to the choice of strategy \( \gamma \in \Gamma \). The problem of finding the optimal mixed strategies can be written as:

\[
\min_{q \in Q(\Gamma)} \sum_{\gamma \in \Gamma} q(\gamma) \tilde{J}(\gamma) \tag{8}
\]

\[
\sum_{\gamma \in \Gamma} q(\gamma) E[\sum_{i=1}^{N} \sum_{t=0}^{T-1} r_s(u_s(t))] \leq R_s \quad s \in [1 \ldots S] \tag{9}
\]

\[
\sum_{\gamma \in \Gamma} q(\gamma) = 1 \tag{10}
\]

where we have one constraint for each of the \( S \) sensor resource pools and an additional simplex constraint in Eq. (10) which ensures that \( q \in Q(\Gamma) \) forms a valid probability distribution. This is a large linear program (LP), where the number of possible variables are the strategies in \( \Gamma \). However, the total number of constraints is \( S+1 \), which establishes that optimal solutions of this LP are mixtures of no more than \( S+1 \) strategies. Thus, one can use a column generation approach [22], [23], [24] to quickly identify an optimal mixed strategy. In this approach, one solves Eq. (8) and Eq. (9) restricting the mixed strategies to be mixtures of a small subset \( \Gamma' \subset \Gamma \). The solution of the restricted LP has optimal dual prices \( \lambda_s, s = 1, \ldots, S \). Using these prices, one can determine a corresponding optimal pure strategy by minimizing (7), which the results in [21] show can be decoupled into \( N \) independent optimization problems, one for each location. Each of these problems is solved as a POMDP using standard algorithms such as point-based value iteration (PBVI) [25] to determine the best pure strategy \( \gamma_i \) for these prices. If the best pure strategy \( \gamma_i \) is already in the set \( \Gamma' \), then the solution of Eq. (8) and Eq. (9) restricted to \( Q(\Gamma') \) is an optimal mixed strategy over all of \( Q(\Gamma) \).

Otherwise, the strategy \( \gamma_i \) is added to the admissible set \( \Gamma' \), and the iteration is repeated. The result is a set of mixed strategies that achieve a performance level that is a lower bound on the original SM optimization problem with hard constraints.

### III. COLUMN GENERATION AND POMDP SUBPROBLEM EXAMPLE

We present an example to illustrate the column generation algorithm and POMDP algorithms discussed previously. In this simple example we consider 100 objects (\( N=100 \)), 2 possible object types (\( D=2 \)) with \( X = \{ \text{non-military vehicle, military vehicle}, \} \), and 2 sensors that each have one mode \( (S = 2 \) and \( M_s = 1 \forall s \in \{1,2\}) \). Sensor \( s \) actions have resource costs: \( r_s \), where \( r_1 = 1, r_2 = 2 \). Sensors return 2 possible observation values, corresponding to binary object classifications, with likelihoods:

\[
P(y_{i,1}(t)|x_i, u_1(t)) \quad P(y_{i,2}(t)|x_i, u_2(t))
\]

\[
\begin{pmatrix}
0.90 & 0.10 \\
0.10 & 0.90
\end{pmatrix}
\]

where the \((j,k)\) matrix entry denotes the likelihood that \( y = j \) if \( x_i = k \). The second sensor has 2% better performance than the first sensor but requires twice as many resources to use. Each sensor has \( R_s = 100 \) units of resources, and can view each location. Each of the 100 locations has a uniform prior of \( \pi_i = [0.5 \ 0.5]^T \). For the performance objective, we use \( c(x_i, u_j) = 1 \) if \( x_i \neq v_j \), and 0 otherwise, where the cost is 1 unit for a classification error.

Table I demonstrates the column generation solution process. The first three columns are initialized by guessing values of resource prices and obtaining the POMDP solutions, yielding expected costs and expected resource use for each sensor at those resource prices. A small LP is solved to obtain the optimal mixture of the first three strategies \( \gamma_1, \ldots, \gamma_3 \), and a corresponding set of dual prices. These dual prices are used in the POMDP solver to generate the fourth column \( \gamma_4 \), which yields a strategy that is different from that of the first 3 columns. The LP is re-solved for mixtures of the first 4 strategies, yielding new resource prices that are used to generate the next column. This process continues until the solution using the prices after 7 columns yields a strategy that was already represented in a previous column, terminating.
the algorithm. The optimal mixture combines the strategies of the second, fifth and sixth columns. When the master problem converges, the optimal cost, $J^*$, for the mixed strategy is 5.95 units. The resulting policy graphs are illustrated in Fig. 1, where branches up indicate measurements $y = 1$ ('non-military') and down $y = 2$ ('military'). The red and green nodes denote the final decision, $v_i$, for a location. Note that the strategy of column 5 uses only the second sensor, whereas the strategies of columns 2 and 6 use only the first sensor. The mixed strategy allows the soft resource constraints to be satisfied with equality. Table I also shows the resource costs and expected classification performance of each column.

The example illustrates some of the issues associated with the use of soft constraints in the optimization: the resulting solution does not lead to SM strategies that will always satisfy the hard constraints Eq. (2). We address this issue in the subsequent section.

IV. RECEDING HORIZON CONTROL

The column generation algorithm described previously solves the approximate SM problem with "soft" constraints in terms of mixed strategies that, on average, satisfy the resource constraints. However, for control purposes, one must select actual SM actions that satisfy the hard constraints Eq. (2). Furthermore, the solutions of the decoupled POMDPs provide individual sensor schedules for each location that must be interleaved into a single coherent sensor schedule. To address this, we will explore a set of RH algorithms that will convert the mixed strategy solutions discussed in the previous section to actions that satisfy the hard constraints, and limit the computational complexity of the resulting algorithm. The RH algorithms have adjustable parameters whose effects we will explore in simulation.

The RH algorithms start at stage $t$ with an information state/resource state pair, consisting of available information about each location $i = 1, \ldots, N$ represented by the conditional probability vector $\pi_i(t)$ and available sensor resources $R_s(t)$, $s = 1, \ldots, S$. The first step in the algorithms is to solve the SM problem of Eq. (5) starting at stage $t$ to final stage $T$ subject to soft constraints Eq. (6), using the hierarchical column generation / POMDP algorithms to get a set of mixed strategies. We introduce a parameter corresponding to the maximum number of sensing actions per location to control the resulting computational complexity of the POMDP algorithms.

The second step is to select sensing actions to implement at the current stage $t$ from the mixed strategies. These strategies are mixtures of at most $S + 1$ pure strategies, with associated probabilistic weights. We explore three approaches for selecting sensing actions:

- str1: Select the pure strategy with maximum probability.
- str2: Randomly select a pure strategy per location according to the optimal mixture probabilities.
- str3: Select the pure strategy with positive probability that minimizes the expected sensor resource use over all sensors (and leaves resources for use in future stages).

The pure strategies that are selected for each location map the current information sets, $I_i(t)$ for location $i$, into a deterministic sensing action. ‘str1’ and ‘str3’ choose the same pure strategy to use across all locations. Note that there may not be enough sensor resources to execute the selected actions, particularly in the case where the pure strategy with maximum probability is selected. To address this, we rank sensing actions by their expected entropy gain [26]:

$$\text{Gain}(u_s(t)) = \frac{H(\pi_s(t)) - E_y[H(\pi_s(t+1))|y, u_s(t)]}{r_s(u_s(t))}$$

where $E_y[]$ is expectation over the potential measurement value. We schedule sensor actions in order of decreasing expected entropy gain, and perform those actions at stage $t$ that have enough sensor resources to be feasible. We also use the entropy gain algorithm at the very end of a simulation when resources are nearly depleted, and the higher cost sensor modes are no longer feasible.

The measurements collected from the scheduled actions are used to update the information states $\pi_i(t+1)$ using Eq. (3). The resources used by the actions are eliminated from the available resources to compute $R_s(t+1)$ using Eq. (4). The RH algorithm is then executed from the new information state/resource state condition.
TABLE II: Observation likelihoods for different sensor modes with the observation symbols y1, y2 and y3.

<table>
<thead>
<tr>
<th></th>
<th>Search</th>
<th>Low-res</th>
<th>Hi-res</th>
</tr>
</thead>
<tbody>
<tr>
<td>empty</td>
<td>0.92</td>
<td>0.95</td>
<td>0.95</td>
</tr>
<tr>
<td>car</td>
<td>0.08</td>
<td>0.05</td>
<td>0.02</td>
</tr>
<tr>
<td>truck</td>
<td>0.08</td>
<td>0.05</td>
<td>0.02</td>
</tr>
<tr>
<td>military</td>
<td>0.08</td>
<td>0.05</td>
<td>0.02</td>
</tr>
</tbody>
</table>

V. SIMULATION RESULTS

In order to evaluate the relative performance of the different RH algorithms, we performed a set of experiments with simulations. In these experiments, there were 100 locations, each of which could be empty, or have objects of three types, so the possible states of location \( i \) were \( x_i \in \{0, 1, 2, 3\} \), where type 1 represents cars, type 2 trucks, and type 3 military vehicles. Sensors can have several modes: a search mode, a low resolution mode and a high resolution mode. The search mode primarily detects the presence of objects; the low resolution mode can identify cars, but confuses the other two types, whereas the high resolution mode can separate the three types. Observations are modeled as having three possible values. The search mode consumes 0.25 units of resources, whereas the low-resolution mode consumes 1 unit and the high resolution mode 5 units, uniformly for each sensor and location. Table II shows the likelihood functions that were used in the simulations.

Initially, each location has a state with one of two prior probability distributions: \( \pi_i(0) = [0.10, 0.60, 0.20, 0.10]^T \), \( i \in \{1, \ldots, 10\} \) or \( \pi_i(0) = [0.80, 0.12, 0.06, 0.02]^T \), \( i \in \{11, \ldots, 100\} \). Thus, the first 10 locations are likely to contain objects, whereas the other 90 locations are likely to be empty. When multiple sensors are present, they may share some locations in common, and have locations that can only be seen by a specific sensor, as illustrated in Fig. 2.

The cost function used in the experiments, \( c(x_i, v_i) \) is shown in Table III. The parameter \( MD \) represents the cost of a missed detection, and will be varied in the experiments.

Table IV shows simulation results for a search and classify scenario involving 2 identical sensors (with the same visibility), evaluating different versions of the RH control algorithms and with different resource levels. The variable “Horizon” is the total number of sensor actions allowed per location plus one additional action for estimating the location content. The table shows results for different resource levels per sensor, from 30 to 70 units, and displays the lower bound performance computed. The missed detection cost \( MD \) is varied from 1 to 10. The results shown in Table IV represent the average of 100 Monte Carlo simulation runs of the RH algorithms.

The results show that using a longer horizon in the planning improves performance minimally, so that using a RH replanning approach with a short horizon can be used to reduce computation time with limited performance degradation. The results also show that the different RH algorithms have performance close to the optimal lower bound in most cases, with the exception being the case of \( MD = 5 \) with 70 units of sensing resources per sensor. For a horizon 6 plan, the longest horizon studied, the simulation performance is close to that of the associated bound. In terms of which strategy is preferable for converting the mixed strategies, the results of Table IV are unclear. For short planning horizons in the RH algorithms, the preferred strategy appears to be to use the least resources (str3), thus allowing for improvement from replanting. For the longer horizons, there was no significant difference in performance among the three strategies. To illustrate the computational requirements of this scenario (4 states, 3 observations, 2 sensors (6 actions), full sensor-overlap), the number of columns generated by the column generation algorithm to compute a set of mixed strategies was on the order of 10-20 columns for the horizon 6 algorithms, which takes about 60 sec on a 2.2 Ghz, single-core, Intel P4 machine under Linux using C code in ‘Debug’ mode (with 1000 belief-points for PBVI). Memory usage without optimizations is around 3 MB.

There are typically 4-5 planning sessions in a simulation. Profiling indicates that roughly 80% of the computing time is
table of content

**TABLE VII**: Simulation results for 3 homogeneous sensors with partial overlap as shown in Fig. 3.

**TABLE VI**: Comparison of performance bounds with 2 homogeneous sensors vs. 2 heterogeneous sensors.

spent on solving the POMDP problems using PBVI and 15% goes towards (recursively) tracing decision-trees in order to back out (deduce) the expected measurement resources used.

In the next set of experiments, we compare the use of heterogeneous sensors that have different modes available. In these experiments, the 100 locations are guaranteed to have an object, so $x_i = 0$ is not feasible. The prior probability of object type for each location is $\pi_i(0) = [0.070.200.1]^T$. Table V shows the results of experiments with sensors that have all sensing modes, versus an experiment where one sensor has only a low-resolution mode and the other sensor has both high and low-resolution modes. The table shows the lower bounds predicted by the column generation algorithm, to illustrate the change in performance expected from the different architectural choices of sensors. The results indicate that specialization of one sensor can lead to significant degradation in performance due to inefficient use of its resources.

The next set of results explore the effect of spatial distribution of sensors. We consider experiments where there are two homogeneous sensors which have only partially-overlapping coverage zones. We define a ‘visibility group’ as a set of sensors that have a common coverage zone. Table VI gives bounds for different percentages of overlap. Note that, even when there is only 20% overlap, the achievable performance is similar to that of the 100% overlap case in Table V, indicating that proper choice of strategies can lead to efficient sharing of resources from different sensors and equalizing their workload. The last set of results show the performance of the RH algorithms for three homogeneous sensors with partial overlap and different resource levels. The visibility groups are graphically portrayed in Fig. 3. Table VII presents the simulated cost values averaged over 100 simulations of the different RH algorithms and the lower bounds. The results support our previous conclusions: when a short horizon is used in the RH algorithm, and there are sufficient resources, the strategy that uses the least resources is preferred as it allows for replanting when new information is available. If the RH algorithm uses a longer horizon, then its performance approaches the theoretical lower bound, and the difference in performance between the three approaches for sampling the mixed strategy to obtain a pure strategy is statistically insignificant. Our results suggest that RH control with modest horizons of 2 or 3 sensor actions per location can yield performance close to the best achievable performance using mixed strategies. If shorter horizons are used to reduce computation, then an approach that samples mixed strategies by using the smallest amount of resources is preferred. The results also show that, with proper SM, geographically distributed sensors with limited visibility can be coordinated to achieve equivalent performance to centrally pooled resources.

In terms of the computational complexity of our RH algorithms, the main bottleneck is the solution of the POMDP problems. The LPs solved in the column generation approach
are small and are solved in minimal time. Solving the POMDPs required to generate each column (one POMDP for each visibility group in cases with partial sensor overlap) is tractable by virtue of the hierarchical breakdown of the SM problem into independent subproblems. It is also possible to accelerate these computations using multi-core CPU or GPU processors, as the POMDPs are highly parallelizable.

VI. CONCLUSIONS

In this paper, we introduce RH algorithms for near-optimal, closed-loop SM with multi-modal, resource-constrained, heterogeneous sensors. These RH algorithms exploit a lower bound formulation developed in earlier work that decomposes the SM optimization into a master problem, which is addressed with linear-programming techniques, and single location stochastic control problems that are solved using POMDP algorithms. The resulting algorithm generates mixed strategies for sensor plans, and the RH algorithms convert these mixed strategies into sensor actions that satisfy sensor resource constraints.

Our simulation results show that the RH algorithms achieve performance close to that of the theoretical lower bounds in [21]. The results also highlight the different benefits of choosing a longer horizon for RH strategies and alternative approaches at sampling the mixed strategy solutions. Our simulations also show the effects of geographically distributing sensors so that there is limited overlap in field of view and the effects of specializing sensors by using a restricted number of modes.

There are many interesting directions for extensions to this work. First, one could consider the presence of object dynamics, where objects can arrive at or depart from specific locations. Second, one could consider a set of objects that have deterministic, but time-varying, visibility profiles. Finally, one could consider approaches that reduce the computational complexity of the resulting algorithms, either through exploitation of parallel computing architectures or through the use of off-line learning or other approximation techniques.

REFERENCES