Adaptive Information Gathering via Imitation Learning

Sanjiban Choudhury*, Ashish Kapoor†, Gireeja Ranade†, Sebastian Scherer* and Debadeepta Dey†
*The Robotics Institute
Carnegie Mellon University, Pittsburgh, PA 15213
Email: {sanjibac,basti}@andrew.cmu.edu
†Microsoft Research, Redmond USA
Email: {akapoor,giranade,dedey}@microsoft.com

***Abstract*—In the adaptive information gathering problem, a policy is required to select an informative sensing location using the history of measurements acquired thus far. While there is an extensive amount of prior work investigating effective practical approximations using variants of Shannon's entropy, the efficacy of such policies heavily depends on the geometric distribution of objects in the world. On the other hand, the principled approach of employing online POMDP solvers is rendered impractical by the need to *explicitly* sample online from a posterior distribution of world maps.**

We present a novel data-driven imitation learning framework to efficiently train information gathering policies. The policy imitates a *clairvoyant oracle* - an oracle that at train time has full knowledge about the world map and can compute maximally informative sensing locations. We analyze the learnt policy by showing that offline imitation of a clairvoyant oracle is *implicitly* equivalent to online oracle execution in conjunction with posterior sampling. This observation allows us to obtain powerful near-optimality guarantees for information gathering problems possessing an adaptive sub-modularity property. As demonstrated on a spectrum of 2D and 3D exploration problems, the trained policies enjoy the best of both worlds - they adapt to different world map distributions while being computationally inexpensive to evaluate.

I. Introduction

This paper examines the following information gathering problem - given a hidden world map, sampled from a prior distribution, the goal is to successively visit sensing locations such that the amount of relevant information uncovered is maximized while not exceeding a specified fuel budget. This problem fundamentally recurs in mobile robot applications such as autonomous mapping of environments using ground and aerial robots [2, 12], monitoring of water bodies [13] and inspecting models for 3D reconstruction [16, 15].

The nature of "interesting" objects in an environment and their spatial distribution influence the optimal trajectory a robot might take to explore the environment. As a result, it is important that a robot learns about the type of environment it is exploring as it acquires more information and adapts its exploration trajectories accordingly. This adaptation must be done online, and we provide such algorithms in this paper.

Consider a robot equipped with a sensor (RGBD camera) that needs to generate a map of an unknown environment. It is given a prior distribution about the geometry of the world - such as a distribution over narrow corridors, or bridges or power-lines. At every time step, the robot visits a sensing location and receives a sensor measurement (e.g. depth image) that has some amount of information utility (e.g. surface coverage of objects with point cloud). If the robot employs a non-adaptive lawnmower-coverage pattern, it can potentially waste time and fuel visiting ineffective locations. On the other hand, if it uses the history of measurements to infer the geometry of unexplored space, it can plan efficient information-rich trajectories. This incentivizes training policies to optimally gather information on the distribution of worlds the robot expects to encounter.

Even though its is natural to think of this problem setting as a POMDP, we frame this problem as a novel data-driven imitation learning problem [33]. We propose a framework that trains a policy on a dataset of worlds by imitating a *clairvoyant oracle*. During the training process, the oracle has full knowledge about the world map (hence clairvoyant) and visits sensing locations that would maximize information. The policy is then trained to imitate these movements as best as it can using partial knowledge from the current history of movements and measurements. As a result of our novel formulation, we are able to sidestep a number of challenging issues in POMDPs like explicitly computing posterior distribution over worlds and planning in belief space.

Our contributions are as follows:

1) We map the information gathering problem to a POMDP and present an approach to solve it using imitation learning of a clairvoyant oracle.
2) We present a framework to train such a policy on the non i.i.d distribution of states induced by the policy itself.
3) We analyze the learnt policy by showing that offline imitation of a clairvoyant oracle is equivalent to online oracle execution in conjunction with posterior sampling.
4) We present results on a variety of datasets to demonstrate the efficacy of the approach.

The remainder of this paper is organized as follows. Section II presents the formal problem and Section III maps it to the imitation learning framework. The algorithms and analysis is presented in Section IV. Section V presents experimental results with conclusions in Section VI.

II. Background

A. Notation

Let $\mathcal{V}$ be a set of nodes corresponding to all sensing locations. The robot starts at node v_s. Let $\xi = (v_1, v_2, \ldots, v_p)$ be a sequence of nodes (a path) such that $v_1 = v_s$. Let Ξ be the

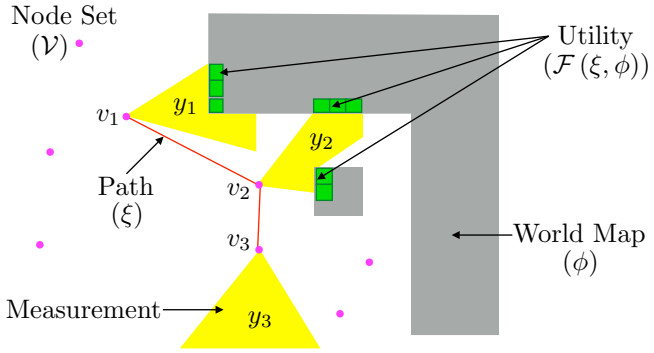


Fig. 1: The adaptive information gathering problem. Given a world map ϕ, the robot plans a path ξ which visits a node $v_i \in \mathcal{V}$ and receives measurement y_i, such that information gathered (utility) $\mathcal{F}(\xi, \phi)$ is maximized.

set of all such paths. Let $\phi \in \mathcal{M}$ be the world map. Let $y \in \mathcal{Y}$ be a measurement received by the robot. Let $\mathcal{H} : \mathcal{V} \times \mathcal{M} \to \mathcal{Y}$ be a measurement function. When the robot is at node v in a world map ϕ, the measurement y received by the robot is $y = \mathcal{H}(v, \phi)$.

Let $\mathcal{F} : 2^{\mathcal{V}} \times \mathcal{M} \to \mathbb{R}_{\geq 0}$ be a utility function. For a path ξ and a world map ϕ, $\mathcal{F}(\xi, \phi)$ assigns a utility to executing the path on the world. We assume the utility function is a set function, i.e. invariant to the sequence of nodes in the path. We further assume the utility function satisfies the property of *adaptive submodularity* - a natural diminishing returns property which we will leverage in the approach we propose. This property is true for several functions such as set coverage function [9]. We assume that the measurement and utility function is deterministic. [1]

Given a node $v \in \mathcal{V}$, a set of nodes $V \subseteq \mathcal{V}$ and world ϕ, the discrete derivative of the utility function $\mathcal{F}$ is $\Delta_{\mathcal{F}}(v \mid V, \phi) = \mathcal{F}(V \cup \{v\}, \phi) - \mathcal{F}(V, \phi)$ Let $\mathcal{T} : \Xi \times \mathcal{M} \to \mathbb{R}_{\geq 0}$ be a travel cost function. For a path ξ and a world map ϕ, $\mathcal{T}(\xi, \phi)$ assigns a travel cost for executing the path on the world. Fig. 1 shows an illustration.

B. Problems with Known World Maps

We define four variants of the information gathering problem. For the first two variants, the world map ϕ is known and can be evaluated while computing a path ξ.

Problem 1 (KNOWN-UNC: Known World Map; Unconstrained Travel Cost)**.** *Given a world map ϕ and a time horizon T, find a path ξ that maximizes utility*

$$\begin{aligned} \underset{\xi \in \Xi}{\arg\max} \quad & \mathcal{F}(\xi, \phi) \\ \textit{s.t.} \quad & |\xi| \leq T + 1 \end{aligned} \tag{1}$$

Problem 2 (KNOWN-CON: Known World Map; Constrained Travel Cost)**.** *Problem 1 with a travel cost budget B*

$$\begin{aligned} \underset{\xi \in \Xi}{\arg\max} \quad & \mathcal{F}(\xi, \phi) \\ \textit{s.t.} \quad & \mathcal{T}(\xi, \phi) \leq B \\ & |\xi| \leq T + 1 \end{aligned} \tag{2}$$

[1]The deterministic assumption implies that there is no utility in visiting a node twice which simplifies computation. We also see that this assumption is applied in practice [16].

Problem 1 is a set function maximization problem which in general can be NP-Hard (Krause and Golovin [23]). However, the utility function $\mathcal{F}$ is a *monotone submodular* function. For such functions, it has been shown that greedy strategies achieve near-optimality (Krause et al. [25], Krause and Guestrin [24]).

Problem 2 introduces a routing constraint (due to $\mathcal{T}$) for which greedy approaches can perform arbitrarily poorly. Chekuri and Pal [3], Singh et al. [35] propose a quasi-polynomial time recursive greedy approach to solving this problem. Iyer and Bilmes [17] solve a related problem (submodular knapsack constraints) using an iterative greedy approach which is generalized by Zhang and Vorobeychik [41]. Yu et al. [40] propose a mixed integer approach to solve a related correlated orienteering problem. Hollinger and Sukhatme [13] propose a sampling based approach. Arora and Scherer [1] use an efficient TSP with random sampling approach.

C. Problems with Hidden World Maps

We now consider the setting where the world map ϕ is hidden. Given a prior distribution $P(\phi)$, it can be inferred only via the measurements y_i received as the robot visits nodes v_i. Hence, instead of solving for a fixed path, we compute a policy that maps history of measurements received and nodes visited to decide which node to visit.

Problem 3 (HIDDEN-UNC: Hidden World Map; Unconstrained Travel Cost)**.** *Given a distribution of world maps, $P(\phi)$, a time horizon T, find a policy that at time t, maps the history of nodes visited $\{v_i\}_{i=1}^{t-1}$ and measurements received $\{y_i\}_{i=1}^{t-1}$ to compute node v_t to visit at time t, such that the expected utility is maximized.*

Problem 4 (HIDDEN-CON: Hidden World Map; Constrained Travel Cost)**.** *Problem 3 with a travel cost budget B*

Due to the hidden world map ϕ, it is not straight forward to apply the approaches discussed in Section II-B - methods have to reason about how $P(\phi \mid \{v_i\}_{i=1}^{t-1}, \{y_i\}_{i=1}^{t-1})$ will evolve. However, we assume the utility function $\mathcal{F}$ has an additional property of *adaptive submodularity* [9]. Hence, applying greedy strategies to Problem 3 has near-optimality guarantees (Golovin et al. [10], Javdani et al. [18, 19], Chen et al. [5, 6]). However, these strategies require explicitly sampling from the posterior distribution of ϕ which make it intractable to apply in this problem.

Problem 4 does not enjoy the adaptive submodularity property. Hollinger et al. [15, 14] propose a heuristic based approach to select a subset of informative nodes and perform minimum cost tours. Singh et al. [36] replan every step using a non-adaptive information path planning algorithm. Inspired by adaptive TSP approaches by Gupta et al. [11], Lim et al. [28, 27] propose recursive coverage algorithms to learn policy trees. However such methods cannot scale well to large state and observation spaces. Heng et al. [12] make a modular approximation of the objective function. Isler et al. [16] survey a broad number of myopic information gain based heuristics that work well in practice but have no formal guarantees.

III. POMDPs and Imitation Learning

A. Mapping Problems to a POMDP

We now map Problems Hidden-Unc and Hidden-Con to a Partially Observable Markov Decision Process (POMDP). The POMDP is a tuple $(\mathcal{S}, \mathcal{M}, \mathcal{A}, \Omega, R, \mathcal{O}, Z, T)$ defined upto a fixed finite horizon T. It is defined over an augmented state space comprising of the ego-motion state space $\mathcal{S}$ (which we will refer to as simply the state space) and the space of world maps $\mathcal{M}$. The first component, $\mathcal{S}$, is fully observable while the second component, $\mathcal{M}$, is partially observable through observations received.

Let the state, $s_t \in \mathcal{S}$, be the set of nodes visited, $s_t = (v_1, v_2, \ldots, v_t)$. [2] Let the action, $a_t \in \mathcal{A}$ be the node visited $a_t = v_{t+1}$. Given a world map ϕ, at state s, the utility of a is $\mathcal{F}(s \cup a, \phi)$. For Problem Hidden-Con, let $\mathcal{A}_{\text{feas}}(s, \phi) \subset \mathcal{A}$ be the set of feasible actions defined as

$$\mathcal{A}_{\text{feas}}(s, \phi) = \{a \mid a \in \mathcal{A}, \mathcal{T}(s \cup a, \phi) \leq B\} \tag{3}$$

The state transition function, $\Omega(s, a, s') = P(s'|s, a)$, is the deterministic function $s' = s \cup a$. The one-step-reward function, $R(s, \phi, a) \in [0, 1]$, is defined as the normalized marginal gain of the utility function, $R(s, \phi, a) = \frac{\Delta_{\mathcal{F}}(a|s,\phi)}{\mathcal{F}(\mathcal{A},\phi)}$. Let the observation, $o_t \in \mathcal{O}$ be the measurement $o_t = y_t$. The observation model, $Z(s, a, \phi, o) = P(o|s, a, \phi)$ is the deterministic function $o = \mathcal{H}(a, \phi)$.

We define the belief, ψ_t, to be the history of state, observation tuple received so far, i.e. $\{(s_i, o_i)\}_{i=1}^{t}$. [3] The belief transition function $P(\psi'|\psi, a)$ can be computed from Ω and Z. Let $\tilde{\pi}(s, \psi) \in \tilde{\Pi}$ be a policy that maps state s and belief ψ to a feasible action $a \in \mathcal{A}_{\text{feas}}(s, \phi)$. The value of executing a policy $\tilde{\pi}$ for t time steps starting at state s and belief ψ is the expected cumulative reward obtained by $\tilde{\pi}$:

$$\tilde{V}_t^{\tilde{\pi}}(s, \psi) = \sum_{i=1}^{t} \mathbb{E}_{\substack{s_i, \psi_i \sim P(s', \psi'|s, \psi, \tilde{\pi}, i), \\ \phi \sim P(\phi|\psi_i)}} \left[R(s_i, \phi, \tilde{\pi}(s_i, \psi_i))\right] \tag{4}$$

where $P(\psi'|\psi, \tilde{\pi}, i)$ is the distribution of beliefs at time i starting from ψ and following policy $\tilde{\pi}$. Similarly $P(s'|s, \tilde{\pi}, i)$ is the distribution of states. $P(\phi|\psi_i)$ is the posterior distribution on worlds given the belief ψ_i.

The state-action value function $\tilde{Q}_t^{\tilde{\pi}}$ is defined as the expected sum of one-step-reward and value-to-go

$$\begin{aligned} \tilde{Q}_t^{\tilde{\pi}}(s, \psi, a) =& \mathbb{E}_{\phi \sim P(\phi|\psi)}\left[R(s, \phi, a)\right] + \\ & \mathbb{E}_{s', \psi' \sim P(s', \psi'|s, \psi, a)}\left[\tilde{V}_{t-1}^{\tilde{\pi}}(s', \psi')\right] \end{aligned} \tag{5}$$

The optimal POMDP policy is obtained by minimization of the expected state-action value function

$$\pi^* = \arg\max_{\tilde{\pi} \in \tilde{\Pi}} \mathbb{E}_{\substack{t \sim U(1:T), \\ s, \psi \sim P(s, \psi|\tilde{\pi}, t)}} \left[\tilde{Q}_{T-t+1}^{\tilde{\pi}}(s, \psi, \tilde{\pi}(s, \psi))\right] \tag{6}$$

[2] We define state s_t to contain all nodes given the utility function is a set function. Refer to [7]

[3] There is a redundancy between state and belief, i.e, state is implicitly captured in belief. This is purely to aid in defining a clairvoyant oracle later on (which needs only the state and disregards the belief). We define belief in a manner such that the posterior distribution of worlds can be computed using ψ_t. In practice, belief can be encoded efficiently by noting $s_{t-1} \subset s_t$.

where $U(1:T)$ is a uniform distribution over the discrete interval $\{1, \ldots, T\}$, $P(s \mid \tilde{\pi}, t)$ and $P(\psi \mid \tilde{\pi}, t)$ are the posterior distribution over states and beliefs following policy $\tilde{\pi}$ for t steps. The value of a policy $\tilde{\pi} \in \tilde{\Pi}$ for T steps on a distribution of worlds $P(\phi)$, starting states $P(s)$ and starting belief $P(\psi|s)$ is the expected value function for the full horizon

$$J(\tilde{\pi}) = \mathbb{E}_{s_1 \sim P(s), \psi_1 \sim P(\psi|s_1)}\left[\tilde{V}_T^{\tilde{\pi}}(s_1, \psi_1)\right] \tag{7}$$

Just as Problems Hidden-Unc and Hidden-Con map to a POMDP, Problems Known-Unc and Known-Con map to the corresponding MDP. While we omit the details for brevity (refer to [7]), the MDP defines a corresponding policy $\pi(s, \phi) \in \Pi$, value function $V_t^{\pi}(s, \phi)$, state-action value function $Q_t^{\pi}(s, \phi, a)$ and optimal policy π_{MDP}.

Online POMDP planning also has a large body of work (see Ross et al. [32]). Although there exists fast solvers such as POMCP (Silver and Veness [34]) and DESPOT (Somani et al. [37]), the product space of world maps and observation space is too large for online planning. An alternative class of approaches is model-free policy improvement ([30]) which has also been applied to POMDPs ([29, 26]). While these methods make very little assumptions about the problem, they are local and require careful initialization.

B. Imitation Learning

An alternative to policy improvement approaches is to train policies to imitate reference policies (or oracle policies). This is a suitable choice for scenarios where the problem objective is to imitate a user-defined policy. This is also a useful approach in scenarios where there exist good oracle policies for the original problem, however these policies cannot be executed online (e.g due to computational complexity) hence requiring imitation via an offline training phase.

We now formally define imitation learning as applied to our setting. Given a policy $\tilde{\pi}$, we define the distribution of states $P(s|\tilde{\pi})$ and beliefs $P(\psi|\tilde{\pi})$ induced by it (termed as *roll-in*). Let $\mathcal{L}(s, \psi, \tilde{\pi})$ be a loss function that captures how well policy $\tilde{\pi}$ imitates an oracle. Our goal is to find a policy $\hat{\pi}$ which minimizes the expected loss as follows.

$$\hat{\pi} = \arg\min_{\tilde{\pi} \in \tilde{\Pi}} \mathbb{E}_{s \sim P(s|\tilde{\pi}), \psi \sim P(\psi|\tilde{\pi})}\left[\mathcal{L}(s, \psi, \tilde{\pi})\right] \tag{8}$$

This is a non-i.i.d supervised learning problem. Ross and Bagnell [33] propose ForwardTraining to train a non-stationary policy (one policy $\hat{\pi}_t$ for each timestep), where each policy $\hat{\pi}_t$ can be trained on distributions induced by previous policies $(\hat{\pi}_1, \ldots, \hat{\pi}_{t-1})$. While this has guarantees, it is impractical given a different policy is needed for each timestep. For training a single policy, Ross and Bagnell [33] show how such problems can be reduced to no-regret online learning using dataset aggregation (DAgger). The loss function they consider $\mathcal{L}$ is a mis-classification loss with respect to what the expert demonstrated. Ross and Bagnell [31] extend the approach to the reinforcement learning setting where $\mathcal{L}$ is

the reward-to-go of an oracle reference policy by aggregating *values* to imitate (AGGREVATE).

C. Solving POMDP via Imitation of a Clairvoyant Oracle

To examine the suitability of imitation learning in the POMDP framework, we compare the training rules (6) and (8). We see that a good candidate loss function $\mathcal{L}(s, \psi, \tilde{\pi})$ should incentivize maximization of $\tilde{Q}^{\tilde{\pi}}_{T-t+1}(s, \psi, \tilde{\pi}(s, \psi))$. A suitable approximation of the optimal value function $\tilde{Q}^{\tilde{\pi}^*}_{T-t+1}$ that can be computed at train time would suffice. In this work we define cumulative reward gathered by a *clairvoyant oracle* as the value function to imitate[4]. This has been shown to be effective by Kahn et al. [20], Sun et al. [39], Karkus et al. [21].

Definition 1 (Clairvoyant Oracle)**.** *Given a distribution of world map $P(\phi)$, a clairvoyant oracle $\pi_{\mathrm{OR}}(s, \phi)$ is a policy that maps state s and world map ϕ to a feasible action $a \in \mathcal{A}_{\mathrm{feas}}(s, \phi)$ such that it approximates the optimal MDP policy, $\pi_{\mathrm{OR}} \approx \pi_{\mathrm{MDP}} = \arg\max_{\pi \in \Pi} J(\pi)$.*

The term *clairvoyant* is used because the oracle has full access to the world map ϕ at train time. The oracle can be used to compute state-action value as follows

$$Q^{\pi_{\mathrm{OR}}}_t(s, \phi, a) = R(s, \phi, a) + \mathbb{E}_{s' \sim P(s'|s,a)} \left[V^{\pi_{\mathrm{OR}}}_{t-1}(s', \phi) \right] \quad (9)$$

Our approach is to imitate the oracle during training. This implies that we train a policy $\hat{\pi}$ by solving the following optimization problem

$$\hat{\pi} = \arg\max_{\tilde{\pi} \in \tilde{\Pi}} \mathbb{E}_{\substack{\phi \sim P(\phi), \\ t \sim U(1:T), \\ s, \psi \sim P(s, \psi | \phi, \tilde{\pi}, t)}} \left[Q^{\pi_{\mathrm{OR}}}_{T-t+1}(s, \phi, \tilde{\pi}(s, \psi)) \right] \quad (10)$$

While we will define training procedures to concretely realize (10) later in Section IV, we offer some intuition behind this approach. Since the oracle π_{OR} knows the world map ϕ, it has appropriate information to assign a value to an action a. The policy $\hat{\pi}$ attempts to imitate this action from the partial information content present in its belief ψ. Due to this realization error, the policy $\hat{\pi}$ visits a different state, updates its belief, gains more information, and queries the oracle for the best action. Hence while the learnt policy can make mistakes in the beginning of an episode, with time it gets better at imitating the oracle.

D. Analysis using a Hallucinating Oracle

The learnt policy imitates a clairvoyant oracle that has access to more information (world map ϕ compared to belief ψ). Hence, the realizability error of the policy is due to two terms - firstly the information mismatch and secondly the expressiveness of feature space. This realizability error can be hard to bound making it difficult to bound the performance of the learnt policy. This motivates us to introduce a hypothetical construct, a *hallucinating oracle*, to alleviate the information mismatch.

Definition 2 (Hallucinating Oracle)**.** *Given a prior distribution of world map $P(\phi)$ and roll-outs by a policy $\tilde{\pi}$, a hallucinating oracle $\tilde{\pi}_{\mathrm{OR}}$ computes the instantaneous posterior distribution over world maps and takes the action with the highest expected state-action value as computed by the clairvoyant oracle.*

$$\tilde{\pi}_{\mathrm{OR}}(s, \psi) = \arg\max_{a \in \mathcal{A}} \mathbb{E}_{\phi' \sim P(\phi'|\psi, \tilde{\pi}, t)} \left[Q^{\pi_{\mathrm{OR}}}_{T-t+1}(s, \phi', a) \right] \quad (11)$$

We will now show that by imitating a clairvoyant oracle we effectively imitate the corresponding hallucinating oracle

Lemma 1. *The **offline** imitation of **clairvoyant** oracle (10) is equivalent to sampling **online** a world from the posterior distribution and executing a **hallucinating** oracle as shown*[5]

$$\hat{\pi} = \arg\max_{\tilde{\pi} \in \tilde{\Pi}} \mathbb{E}_{\substack{\phi \sim P(\phi), \\ t \sim U(1:T), \\ s, \psi \sim P(s, \psi | \phi, \tilde{\pi}, t)}} \left[Q^{\tilde{\pi}_{\mathrm{OR}}}_{T-t+1}(s, \phi, \tilde{\pi}(s, \psi)) \right]$$

Note that a hallucinating oracle uses the same information content as the learnt policy. Hence the realization error is purely due to the expressiveness of the feature space. However, we now have to analyze the performance of the hallucinating oracle which chooses the best action at a time step given its current belief. We will see in Section IV-C that this policy is near-optimal for Problem HIDDEN-UNC. For Problem HIDDEN-CON, while this does not have any such guarantees, there is evidence to show this is an effective policy as alluded to in Koval et al. [22].

IV. APPROACH

A. Overview

We introduced imitation learning and its applicability to POMDPs in Section III. We now present a set of algorithms to concretely realize the process. The overall idea is as follows - we are training a policy $\hat{\pi}(s, \psi)$ that maps features extracted from state s and belief ψ to an action a. The training objective is to imitate a clairvoyant oracle that has access to the corresponding world map ϕ. Fig. 2 shows an abstract overview. In order to define concrete algorithms, there are two degrees of freedom that need to be specified

1) *Clairvoyant oracle to imitate*: The choice of the clairvoyant oracle defines $Q^{\pi_{\mathrm{OR}}}_t(s, \phi, a)$ in (9). Depending on whether we are solving Problem HIDDEN-UNC or HIDDEN-CON, we explore two different kinds of oracles
   (a) *Clairvoyant one-step-reward*: For Problem HIDDEN-UNC, we use the one-step-reward $R(s, \phi, a)$ in place of $Q^{\pi_{\mathrm{OR}}}_t(s, \phi, a)$. This corresponds to greedily imitating the immediate reward. We will see in Section IV-C that this has powerful near-optimality guarantees.
   (b) *Clairvoyant reward-to-go*: For Problem HIDDEN-CON, we define an oracle policy π_{OR} that approximately

[4]Imitation of a clairvoyant oracle in information gathering has been explored by Choudhury et al. [7]. We subsume the presented algorithm, EXPLORE, in our framework (as Algorithm QVALAGG) and instead focus on the theoretical insight on what it means to imitate a clairvoyant oracle. Additionally, we provide analysis highlighting when such an approach is effective (in Section III-D)

[5]Refer to [8] for all proofs

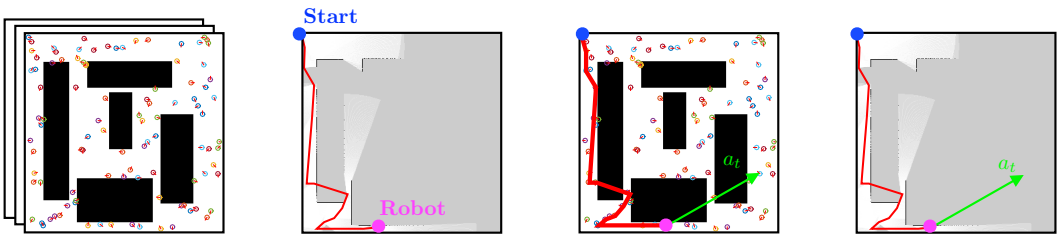


Fig. 2: An abstract overview of the imitation learning architecture where a learner $\hat{\pi}$ is trained to imitate a clairvoyant oracle π_{OR}. There are 4 key steps. Step 1: A world map ϕ is sampled from database representing $P(\phi)$. Step 2: A policy is used to roll-in on ϕ to a timestep t to get state s_t and belief ψ_t. Step 3: A random action a_t is chosen and a clairvoyant oracle π_{OR} is given full access to world map ϕ to compute the cumulative reward to go $Q^{\pi_{\text{OR}}}_{T-t+1}(s_t, \phi, a_t)$ Step 4: The learnt policy $\hat{\pi}$ is updated to map (s_t, ψ_t, a_t) to $Q^{\pi_{\text{OR}}}$.

solves the underlying MDP and use its cumulative reward-to-go to compute $Q^{\pi_{\text{OR}}}_t(s, \phi, a)$ (refer (9)).

2) *Stationary / Non-stationary policy*: As mentioned in Section. III-B, imitation learning deals with non i.i.d distributions induced by the policy themselves. Depending on the choice of policy type (non-stationary / stationary), a corresponding training algorithm exists that offers guarantees
   (a) *Non-stationary policy*: For the non-stationary case, we have a policy for each timestep $\hat{\pi}^1, \ldots, \hat{\pi}^T$. While this can be trained using the FORWARDTRAINING algorithm [33], there are several drawbacks. Firstly, it is impractical to have a different policy for each time-step as it scales with T. Secondly, the training has to proceed sequentially. Thirdly, each policy operates on data for only that time-step, thus preventing generalizations across timesteps. However, the procedure is presented for completeness.
   (b) *Stationary policy*: A single stationary policy $\hat{\pi}$ can be trained using the AGGREVATE algorithm [31] that reduces the problem to no-regret online learning setting. The training procedure is an interactive process where data is aggregated to refine the policy. The advantages are that the policy uses data across all timesteps, only one policy needs to be tracked and the training process can be stopped arbitrarily.

B. Algorithm

We now present concrete algorithms to realize the training procedure. Given the two axes of variation - problem and policy type - we have four possible algorithms

1) REWARDFT: Imitate one-step-reward using non-stationary policy by FORWARDTRAINING (Alg. 1)
2) QVALFT: Imitate reward-to-go using non-stationary policy by FORWARDTRAINING (Alg. 1)
3) REWARDAGG: Imitate one-step-reward using stationary policy by DAGGER (Alg. 2)
4) QVALAGG: Imitate reward-to-go using stationary policy by AGGREVATE (Alg. 2)

Table. I shows the algorithm mapping.

Alg. 1 describes the FORWARDTRAINING procedure to train the non-stationary policy. Previous policies $\hat{\pi}^1, \ldots, \hat{\pi}^{t-1}$ are

TABLE I: Mapping from Problem and Policy type to Algorithm

Policy \ Problem	HIDDEN-UNC	HIDDEN-CON
Non-stationary policy	REWARDFT	QVALFT
Stationary policy	REWARDAGG	QVALAGG

Algorithm 1 Non-stationary policy (REWARDFT, QVALFT)

for $t = 1$ **to** T **do**
  Initialize $\mathcal{D} \leftarrow \emptyset$.
  for $j = 1$ **to** m **do**
    Sample world map ϕ from dataset $P(\phi)$
    Execute policy $\hat{\pi}^1, \ldots, \hat{\pi}^{t-1}$ to reach (s_t, ψ_t).
    Execute any action $a_t \in \mathcal{A}_{\text{feas}}(s_t, \phi)$.
    Execute oracle π_{OR} from $t+1$ to T on ϕ
    Collect value to go $Q^{\pi_{\text{OR}}}_i = Q^{\pi_{\text{OR}}}_{T-t+1}(s_t, \phi, a_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{s_t, \psi_t, a_t, t, Q^{\pi_{\text{OR}}}_i\}$
  Train cost-sensitive classifier $\hat{\pi}^t$ on $\mathcal{D}$
Return Set of policies for each time step $\hat{\pi}^1, \ldots, \hat{\pi}^T$.

Algorithm 2 Stationary policy (REWARDAGG, QVALAGG)

Initialize $\mathcal{D} \leftarrow \emptyset$, $\hat{\pi}_1$ to any policy in $\tilde{\Pi}$
for $i = 1$ **to** N **do**
  Initialize sub dataset $\mathcal{D}_i \leftarrow \emptyset$
  Let roll-in policy be $\pi_{\text{mix}} = \beta_i \pi_{\text{OR}} + (1 - \beta_i)\hat{\pi}_i$
  Collect m data points as follows:
  for $j = 1$ **to** m **do**
    Sample world map ϕ from dataset $P(\phi)$
    Sample uniformly $t \in \{1, 2, \ldots, T\}$
    Assign initial state $s_1 = v_s$
    Execute π_{mix} up to time $t-1$ to reach (s_t, ψ_t)
    Execute any action $a_t \in \mathcal{A}_{\text{feas}}(s_t, \phi)$
    Execute oracle π_{OR} from $t+1$ to T on ϕ
    Collect value-to-go $Q^{\pi_{\text{OR}}}_i = Q^{\pi_{\text{OR}}}_{T-t+1}(s_t, \phi, a_t)$
    $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \{s_t, \psi_t, a_t, t, Q^{\pi_{\text{OR}}}_i\}$
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$
  Train cost-sensitive classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$
  (*Alternately: use any online learner* $\hat{\pi}_{i+1}$ *on* $\mathcal{D}_i$)
Return best $\hat{\pi}_i$ on validation

used to create a dataset (Lines 1–5). The oracle provides the value to imitate (Line 7) - if the one-step-reward is used the algorithm is referred to REWARDFT else if an oracle is used QVALFT.

Alg. 2 describes the AGGREVATE procedure to train the stationary policy. The algorithm iteratively trains a sequence of learnt policies $(\hat{\pi}_1, \hat{\pi}_2, \ldots, \hat{\pi}_N)$ by aggregating data for an online cost-sensitive classification problem. At any given iteration, data is collected by roll-in [6] with a mixture of learnt and oracle policy (Lines 1–10). The oracle provides the value to imitate (Lines 12–13) - if the one-step-reward is used the algorithm is referred to REWARDAGG else if an oracle is used QVALAGG. Data is appended to the original dataset and used to train an updated learner $\hat{\pi}_{i+1}$ (Lines 15–17).

C. Imitation of Clairvoyant One-Step-Reward

The strategy to imitate the clairvoyant one-step-reward is employed in Problem HIDDEN-UNC. This is motivated by the observation that the utility function satisfies a property of adaptive submodularity (as mentioned in Section II-C). For such problems, greedily selecting actions with highest expected reward has near-optimality guarantees. This implies the following Lemma

Lemma 2. *The performance of the hallucinating oracle $\tilde{\pi}_{\mathrm{OR}}$ is near-optimal w.r.t the optimal policy π^*.*

$$J\left(\tilde{\pi}_{\mathrm{OR}}\right) \geq \left(1 - \frac{1}{e}\right) J\left(\pi^*\right)$$

This property can then be used to obtain a near-optimality bound for the learnt policy

Theorem 1. *N iterations of* REWARDAGG, *collecting m regression examples per iteration guarantees that with probability at least $1-\delta$*

$$\begin{aligned} J\left(\hat{\pi}\right) \geq & \left(1 - \frac{1}{e}\right) J\left(\pi^*\right) \\ & - 2\sqrt{|\mathcal{A}|} T \sqrt{\varepsilon_{\mathrm{class}} + \varepsilon_{\mathrm{reg}} + O\left(\sqrt{\log\left(\left(1/\delta\right)/Nm\right)}\right)} \\ & - O\left(\frac{T^2 \log T}{\alpha N}\right) \end{aligned}$$

where $\varepsilon_{\mathrm{reg}}$ is the empirical average online learning regret on the training regression examples collected over the iterations and $\varepsilon_{\mathrm{class}}$ is the empirical regression regret of the best regressor in the policy class.

D. Imitation of Clairvoyant Reward-To-Go

Problem HIDDEN-CON does not posses the adaptive submodularity property. However there is empirical evidence to suggest that the hallucinating oracle performance $J\left(\tilde{\pi}_{\mathrm{OR}}\right)$ is sufficiently high [22]. Hence the learnt policy has a performance guarantee with respect to the hallucinating oracle

[6] We sample t instead of using the entire trajectory to minimize the calls to oracle

Theorem 2. *N iterations of* QVALAGG, *collecting m regression examples per iteration guarantees that with probability at least $1-\delta$*

$$\begin{aligned} J\left(\hat{\pi}\right) \geq & J\left(\tilde{\pi}_{\mathrm{OR}}\right) \\ & - 2\sqrt{|\mathcal{A}|} T \sqrt{\varepsilon_{\mathrm{class}} + \varepsilon_{\mathrm{reg}} + O\left(\sqrt{\log\left(\left(1/\delta\right)/Nm\right)}\right)} \\ & - O\left(\frac{T^2 \log T}{\alpha N}\right) \end{aligned}$$

where $\varepsilon_{\mathrm{reg}}$ is the empirical average online learning regret on the training regression examples collected over the iterations and $\varepsilon_{\mathrm{class}}$ is the empirical regression regret of the best regressor in the policy class.

V. EXPERIMENTAL RESULTS

A. Implementation Details

Our implementation is open sourced for both MATLAB and C++ (https://bitbucket.org/sanjiban/matlab_learning_info_gain). For further details regarding implementation, refer [7].

1) Problem Details: The utility function $\mathcal{F}$ is selected to be a fractional coverage function (similar to [16]) which is defined as follows. The world map ϕ is represented as a voxel grid representing the surface of a 3D model. The sensor measurement $\mathcal{H}\left(v, \phi\right)$ at node v is obtained by ray-casting on this 3D model. A voxel of the model is said to be 'covered' by a measurement received at a node if a point lies in that voxel. The coverage of a path ξ is the fraction of covered voxels by the union of measurements received when visiting each node of the path. The travel cost function $\mathcal{T}$ is chosen to be the euclidean distance. The values of total time step T and travel budget B vary with problem instances and are specified along with the results.

2) Learning Details: The tuple (s, a, ψ) is mapped to a vector of features $f = \begin{bmatrix} f_{\mathrm{IG}}^T & f_{\mathrm{mot}}^T \end{bmatrix}^T$. The feature vector f_{IG} is a vector of information gain metrics as described in [16]. f_{mot} encodes the relative rotation and translation required to visit a node. Random forest regression is used as a function approximator. The oracle used is the generalized cost benefit algorithm (GCB) [41].

3) Baseline: The baseline policies are a class of information gain heuristics discussed in [16] augmented with a motion penalization term when applied to Problem HIDDEN-CON. The heuristics are remarkably effective, however, their performance depends on the distribution of objections in a world map.

B. Adaptation to Different Distributions

We created a set of 2D exploration problems to gain a better understanding of the learnt policies and baseline heuristics. The problem was HIDDEN-UNC, the dataset comprises of 2D binary world maps, uniformly distributed nodes and a simulated laser. The problem details are $T = 30$ and $|\mathcal{A}| = 300$. The train size is 100, test size is 100. REWARDAGG is executed for 10 iterations.

The overall conclusion is that on changing the datasets the performance of the heuristics vary widely while the learnt

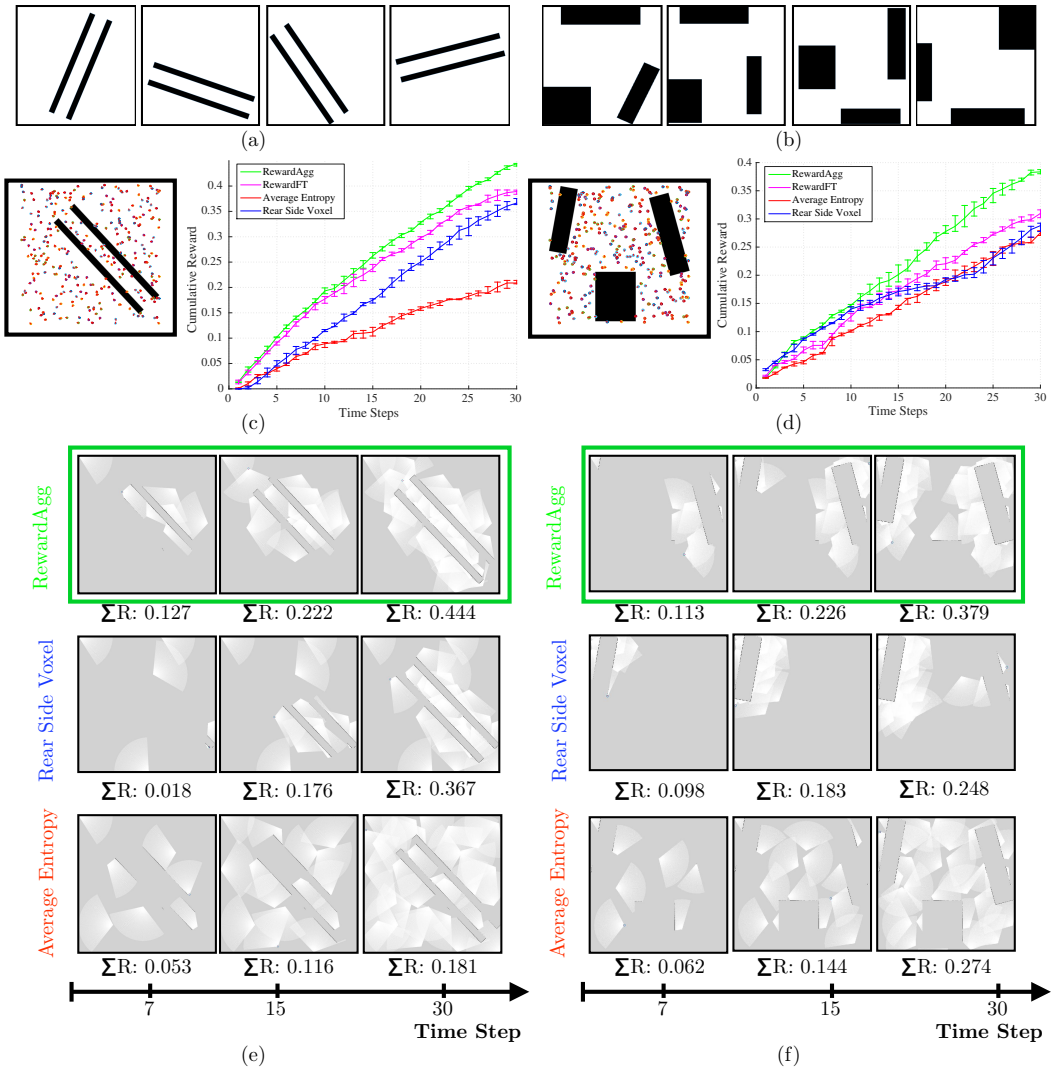


Fig. 3: Case study of Problem HIDDEN-UNC using REWARDAGG, REWARDFT and baseline heuristics. Two different datasets of 2D exploration are considered - (a) dataset 1 (parallel lines) and (b) dataset 2 (distributed blocks). Problem details are: $T = 30, |\mathcal{A}| = 300$, 100 train and 100 test maps. A sample test instance is shown along with a plot of cumulative reward with time steps for different policies is shown in (c) and (d). The error bars show 95% confidence intervals. (e) and (f) show snapshots of the execution at time steps $7, 15$ and 30.

policies outperform both heuristics. Interstingly, REWARDAGG outperforms REWARDFT- this is probably due to the generalization across time-steps. [7] We now analyze each dataset.

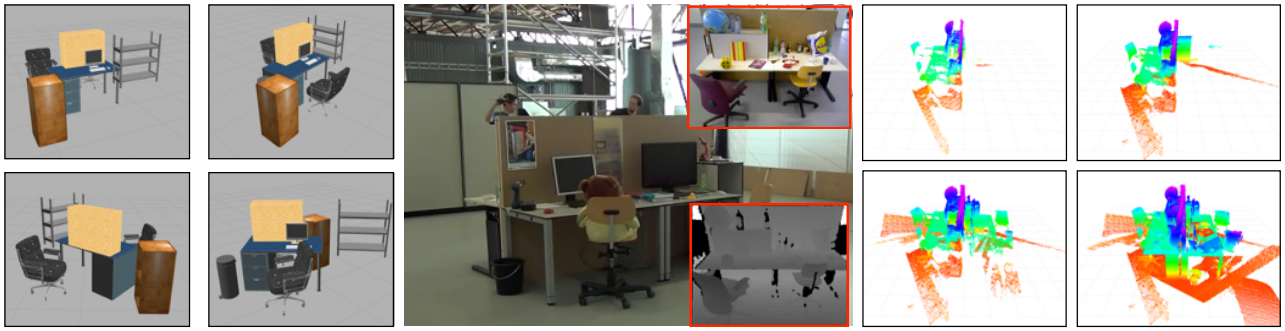
1) Dataset 1: Parallel Lines: Fig. 3a shows a dataset created by applying random affine transformations to a pair of parallel lines. This dataset is representative of information being concentrated in a particular fashion. Fig. 3c shows a comparison of REWARDAGG, REWARDFT with baseline heuristics. While Rear Side Voxel outperforms Average Entropy, REWARDAGG outperforms both. Fig. 3e shows progress of each. Average Entropy explores the whole world without focusing, Rear Side Voxel exploits early while QVALAGG trades off exploration and exploitation.

2) Dataset 2: Distributed Blocks: Fig. 3b shows a dataset created by randomly distributing rectangular blocks around the periphery of the map. This dataset is representative of information being distributed around. Fig. 3c shows that Rear Side Voxel saturates early, Average Entropy eventually overtaking it while REWARDAGG outperforms all. Fig. 3e shows that Rear Side Voxel gets stuck exploiting an island of information. Average Entropy takes broader sweeps of the area thus gaining more information about the world. QVALAGG shows a non-trivial behavior exploiting one island before moving to another.

C. Train on Synthetic, Test on Real

To show the practical usage of our pipeline, we show a scenario where a policy is trained on synthetic data and tested on a real dataset. Fig. 4a shows some sample worlds created in Gazebo to represent an office desk environment on which QVALAGG is trained. Fig. 4b shows a dataset of an office desk collected by TUM Computer Vision Group [38]. The dataset is parsed to create a pair of pose and registered point cloud which can then be used to evaluate different algorithms.

[7] However, we believe given enough data, the non-stationary policy would eventually outperform the stationary policy.



(a) (b) (c)

Fig. 4: Comparison of QVALAGG with baseline heuristics on a 3D exploration problem where training is done on simulated world maps and testing is done on a real dataset of an office workspace. The problem details are: $T = 10$, $B = 12$, $|\mathcal{A}| = 50$. (a) Samples from 100 simulated worlds resembling an office workspace created in Gazebo. (b) Real dataset collected by [38] using a RGBD camera. (c) Snapshots of execution of QVALAGG heuristic at time steps $1, 3, 5, 9$.

Dataset	Sample World Maps	Problem	RewardAgg / QvalAgg	Average Entropy	Occlusion Aware	Unobserved Voxels	Rear Side Voxels	Rear Side Entropy
Poisson Forest of Circular Discs (2D)		HIDDEN-UNC	(0.58, 0.61)	**(0.59, 0.62)**	(0.49, 0.53)	(0.39, 0.45)	(0.53, 0.55)	(0.42, 0.47)
		HIDDEN-CON	**(0.54, 0.59)**	**(0.54, 0.59)**	(0.42, 0.46)	(0.34, 0.41)	(0/37, 0.43)	(0.39, 0.44)
Tabular World of Rectilinear Blocks (2D)		HIDDEN-UNC	**(0.43, 0.53)**	(0.31, 0.35)	(0.20, 0.26)	(0.28, 0.35)	(0.35, 0.44)	(0.25, 0.31)
		HIDDEN-CON	**(0.27, 0.33)**	(0.26, 0.29)	(0.18, 0.23)	(0.21, 0.28)	(0.18, 0.24)	(0.21, 0.27)
Bookshelves and Tables (3D)		HIDDEN-UNC	**(0.14, 0.31)**	(0.01, 0.04)	(0.01, 0.04)	(0.01, 0.04)	(0.01, 0.22)	(0.01, 0.19)
		HIDDEN-CON	**(0.05, 0.24)**	(0.01, 0.04)	(0.01, 0.04)	(0.01, 0.04)	(0.01, 0.22)	(0.01, 0.19)
Cluttered Construction Site (3D)		HIDDEN-UNC	**(0.14, 0.20)**	(0.01, 0.12)	(0.01, 0.09)	(0.01, 0.09)	(0.01, 0.11)	(0.01, 0.10)
		HIDDEN-CON	**(0.08, 0.12)**	(0.01, 0.12)	(0.01, 0.09)	(0.01, 0.09)	(0.01, 0.11)	(0.01, 0.10)
Office Desk and Chairs (3D)		HIDDEN-UNC	**(0.69, 0.80)**	(0.46, 0.59)	(0.51, 0.63)	(0.51, 0.63)	(0.59, 0.67)	(0.61, 0.72)
		HIDDEN-CON	**(0.55, 0.72)**	(0.46, 0.59)	(0.48, 0.63)	(0.48, 0.63)	(0.43, 0.52)	(0.41, 0.53)

Fig. 5: Results for Problems HIDDEN-UNC and HIDDEN-CON on a spectrum of 2D and 3D exploration problems. The train size is 100 and test size is 10. Numbers are the confidence bounds (for 95% CI) of cumulative reward at the final time step. Algorithm with the highest median performance is emphasized in bold.

Fig. 4c shows QVALAGG learns a desk exploring policy by circumnavigating around the desk. This shows the powerful generalization capabilities of the approach.

D. Spectrum of 2D / 3D exploration problems

We evaluate the framework on a spectrum of 2D / 3D exploration problems on synthetic worlds as shown in Fig. 5. For Problem HIDDEN-UNC, REWARDAGG is employed along with baseline heuristics. For Problem HIDDEN-CON, QVALAGG is employed with baseline heuristic augmented with motion penalization. The train size is 100 and test size is 10. We see that the learnt policies outperform all heuristics on most datasets by exploiting the distribution of objects in the world particularly in Problem HIDDEN-UNC. This is indicative of the power of the hallucinating oracle. However, the Poisson forest datasets stand out - where given the ergodic distribution, the Average Entropy heuristic performs best.

VI. CONCLUSION

We present a novel framework for learning information gathering policies via imitation learning of clairvoyant oracles. We presented analysis that establishes an equivalence to online imitation of hallucinating oracles thereby explaining the success of such policies. The framework is validated on a spectrum of 2D and 3D exploration problems.

There are several key directions for future research. Firstly, analysis from Chen et al. [4], where the authors show that under mild assumptions POMDPs can be reduced to sequence of MDPs, is promising for obtaining better regret guarantees. [27] also offer alternatives to adaptive submodularity that could improve guarantees for HIDDEN-CON. Secondly, instead of learning a policy, learning a surrogate utility function that is solved online with a routing TSP might lead to improved performance. Finally, we are looking to apply our framework to POMDP problems where hindsight optimization have shown success (e.g learning to grasp Koval et al. [22]).

VII. Acknowledgement

The authors thank Sankalp Arora for insightful discussions and open source code for exploration in MATLAB.

References

[1] Sankalp Arora and Sebastian Scherer. Randomized Algorithm for Informative Path Planning with Budget Constraints. In *ICRA*, 2017.

[2] Benjamin Charrow, Gregory Kahn, Sachin Patil, Sikang Liu, Ken Goldberg, Pieter Abbeel, Nathan Michael, and Vijay Kumar. Information-Theoretic Planning with Trajectory Optimization for Dense 3D Mapping. In *RSS*, 2015.

[3] Chandra Chekuri and Martin Pal. A recursive greedy algorithm for walks in directed graphs. In *FOCS*, 2005.

[4] Min Chen, Emilio Frazzoli, David Hsu, and Wee Sun Lee. POMDP-lite for Robust Robot Planning under Uncertainty. *arXiv:1602.04875*, 2016.

[5] Yuxin Chen, Shervin Javdani, Amin Karbasi, J. Bagnell, Siddhartha Srinivasa, and Andreas Krause. Submodular Surrogates for Value of Information. In *AAAI*, 2015.

[6] Yuxin Chen, S. Hamed Hassani, and Andreas Krause. Near-optimal bayesian active learning with correlated and noisy tests. *CoRR*, abs/1605.07334, 2016.

[7] Sanjiban Choudhury, Ashish Kapoor, Gireeja Ranade, and Debadeepta Dey. Learning to Gather Information via Imitation. In *ICRA*, 2017.

[8] Sanjiban Choudhury, Ashish Kapoor, Gireeja Ranade, and Debadeepta Dey. Adaptive Information Gathering via Imitation Learning. *arXiv:1705.07834*, 2017.

[9] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *JAIR*, 2011.

[10] Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In *NIPS*, 2010.

[11] Anupam Gupta, Viswanath Nagarajan, and R Ravi. Approximation algorithms for optimal decision trees and adaptive TSP problems. In *International Colloquium on Automata, Languages, and Programming*, 2010.

[12] Lionel Heng, Alkis Gotovos, Andreas Krause, and Marc Pollefeys. Efficient visual exploration and coverage with a micro aerial vehicle in unknown environments. In *ICRA*, 2015.

[13] Geoffrey A Hollinger and Gaurav S Sukhatme. Sampling-based Motion Planning for Robotic Information Gathering. In *RSS*, 2013.

[14] Geoffrey A Hollinger, Brendan Englot, Franz S Hover, Urbashi Mitra, and Gaurav S Sukhatme. Active planning for underwater inspection and the benefit of adaptivity. *IJRR*, 2012.

[15] Geoffrey A Hollinger, Urbashi Mitra, and Gaurav S Sukhatme. Active Classification: Theory and Application to Underwater Inspection. In *Robotics Research*, pages 95–110. Springer, 2017.

[16] Stefan Isler, Reza Sabzevari, Jeffrey Delmerico, and Davide Scaramuzza. An Information Gain Formulation for Active Volumetric 3D Reconstruction. In *ICRA*, 2016.

[17] Rishabh K Iyer and Jeff A Bilmes. Submodular optimization with submodular cover and submodular knapsack constraints. In *NIPS*, 2013.

[18] Shervin Javdani, Matthew Klingensmith, J. Andrew Bagnell, Nancy Pollard, and Siddhartha Srinivasa. Efficient Touch Based Localization through Submodularity. In *ICRA*, 2013.

[19] Shervin Javdani, Yuxin Chen, Amin Karbasi, Andreas Krause, J. Andrew Bagnell, and Siddhartha Srinivasa. Near Optimal Bayesian Active Learning for Decision Making. In *AISTATS*, 2014.

[20] Gregory Kahn, Tianhao Zhang, Sergey Levine, and Pieter Abbeel. PLATO: Policy Learning using Adaptive Trajectory Optimization, 2017.

[21] Peter Karkus, David Hsu, and Wee Sun Lee. QMDP-Net: Deep Learning for Planning under Partial Observability. *arXiv preprint arXiv:1703.06692*, 2017.

[22] Michael Koval, Nancy Pollard, and Siddhartha Srinivasa. Pre- and Post-Contact Policy Decomposition for Planar Contact Manipulation Under Uncertainty. In *RSS*, 2014.

[23] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 2012.

[24] Andreas Krause and Carlos Guestrin. Near-optimal Observation Selection Using Submodular Functions. In *AAAI*, 2007.

[25] Andreas Krause, Jure Leskovec, Carlos Guestrin, Jeanne VanBriesen, and Christos Faloutsos. Efficient sensor placement optimization for securing large water distribution networks. *Journal of Water Resources Planning and Management*, 2008.

[26] Hui Li, Xuejun Liao, and Lawrence Carin. Multi-task reinforcement learning in partially observable stochastic environments. *Journal of Machine Learning Research*, 2009.

[27] Zhan Wei Lim, David Hsu, and Wee Sun Lee. Adaptive Stochastic Optimization: From Sets to Paths. In *NIPS*. 2015.

[28] Zhan Wei Lim, David Hsu, and Wee Sun Lee. Adaptive informative path planning in metric spaces. *IJRR*, 2016.

[29] Miao Liu, Xuejun Liao, and Lawrence Carin. Online Expectation Maximization for Reinforcement Learning in POMDPs. In *IJCAI*, 2013.

[30] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *IROS*, 2006.

[31] Stephane Ross and J Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv:1406.5979*, 2014.

[32] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for POMDPs. *JAIR*, 2008.

[33] Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In *AISTATS*, volume 1, page 6, 2011.

[34] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *NIPS*, 2010.

[35] Amarjeet Singh, Andreas Krause, Carlos Guestrin, William Kaiser, and Maxim Batalin. Efficient Planning of Informative Paths for Multiple Robots. In *IJCAI*, 2007.

[36] Amarjeet Singh, Andreas Krause, and William J. Kaiser. Nonmyopic Adaptive Informative Path Planning for Multiple Robots. In *IJCAI*, 2009.

[37] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. In *NIPS*, 2013.

[38] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *IROS*, 2012.

[39] Wen Sun, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell. Deeply AggreVaTeD: Differentiable Imitation Learning for Sequential Prediction. In *ICML*, 2017.

[40] Jingjin Yu, Mac Schwager, and Daniela Rus. Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks. In *IROS*, 2014.

[41] Haifeng Zhang and Yevgeniy Vorobeychik. Submodular Optimization with Routing Constraints, 2016.