Selecting Operator Queries using Expected Myopic Gain

Robert Cohn, Michael Maxim, Edmund Durfee, and Satinder Singh
Computer Science and Engineering
University of Michigan
Ann Arbor, USA
{*rwcohn, mmaxim, durfee, baveja*}*@umich.edu*

Abstract

When its human operator cannot continuously supervise (much less teleoperate) an agent, the agent should be able to recognize its limitations and ask for help when it risks making autonomous decisions that could significantly surprise and disappoint the operator. Inspired by previous research on making exploration-exploitation tradeoff decisions and on inverse reinforcement learning, we develop Expected Myopic Gain (EMG), a Bayesian approach where an agent explicitly models its uncertainty and how possible operator responses to queries could improve its decisions. With EMG, an agent can weigh the relative expected utilities of seeking operator help versus acting autonomously. We provide conditions under which EMG is optimal, and preliminary empirical results on simple domains showing that EMG can perform well even when its optimality conditions are violated.

***Keywords*- Human-robot/agent Interaction, Planning, Value of Information**

I. Introduction

We consider a single agent (e.g., a robotic vehicle [1]) acting on behalf of a human operator in a sequential decision-making environment modeled as a Markov Decision Process (MDP). The agent knows only the aspects of the environment and operator preferences that the operator has given it, expecting that more details can be provided as needed. However, fatigue, distraction, and other attentional demands mean the operator cannot reliably volunteer, unprompted, such details. Thus, the focus of this paper is on allowing the agent itself to rationally determine whether to ask for help (and if so, what to ask) or to act autonomously.

Our research extends prior work on value of perfect information for exploration-exploitation (EE). Like other methods (e.g. [2]), our agent models its incomplete knowledge about the MDP using a Bayesian representation via parametric probability distributions over possible MDPs. Bayesian Inverse Reinforcement Learning [3] focuses on improving the distribution over the MDPs based on sample optimal behavior (such as during teleoperation). The adapted distribution leads to an improved policy for the agent. In contrast, in EE research the agent itself decides what and when to learn when it chooses an action, since the consequences of taking an action in its current state provides the agent both exploration value (a more informed distribution over MDPs) as well as exploitation value (an immediate reward). Since future action selections will also have exploration and exploitation values, a theoretically powerful EE approach is to solve such a Bayes-Adaptive MDP (e.g., [4]) for a policy that selects the action in each information-state that optimizes the long-term ability to explore and exploit. Unfortunately, doing so is computationally intractable in all but the simplest cases. A useful approximation (see [2]) optimizes long-term exploitation but only short-term ("myopic") exploration by assuming that the distribution over MDPs will only be changed by the single current action choice, and that any resulting change in policy applies forever afterward. Our research adopts this "myopic" approximation, where our agent picks the *querying* action that, in expectation, will provide the highest value of information for guiding current and future action choices.

This paper's main contribution is the Expected Myopic Gain (EMG) algorithm that an agent can use to compute the expected gain in long-term value of asking the operator a query. Like the value of information calculation in [2], EMG is myopic in exploration but long-term in exploitation. EMG extends this concept to allow the agent to evaluate the gain of knowing any aspect of its MDP distribution (rather than just the optimal action for the current state), and applies this capability to decide online whether and what to ask an operator who is able to answer such queries. In the following, we begin by explaining more fully the background ideas EMG builds on, then move on to describe the details of EMG and conditions under which it is optimal, and show preliminary empirical results confirming that EMG can perform well even when its optimality conditions are compromised. We then discuss methods for scaling EMG to larger problems, and end by contrasting EMG with other selective querying approaches.

II. Background on Bayesian MDPs

MDPs: The agent's sequential decision making environment is modeled as an MDP M defined by a tuple $\langle S, A, T, R \rangle$: at each time step the agent observes the state $s \in S$ of the environment, takes an action $a \in A$

which transitions the environment to a random next state $s' \in S$ chosen from the multinomial transition probability distribution $T(s,a)$, and receives a random reward chosen from $R(s')$ (a multinomial over a discrete set of real values). For a policy $\pi : S \rightarrow A$, the long-term value of behaving according to π from state s in MDP M is denoted $V_M^\pi(s)$ and is the expected sum of discounted rewards obtained by the agent when it behaves according to π from start state s. The optimal policy π^* achieves the maximum value among all policies π, i.e., $\pi^* \in \arg\max_\pi V_M^\pi$ (in MDPs, there always exists a policy that simultaneously maximizes the value of every state). The value function for policy π^* is denoted $V_M^{\pi^*}$. Finally, algorithms from dynamic programming, e.g., value iteration or policy iteration, or from linear programming, can be used to find the optimal policy for an MDP.

Bayesian MDPs: An agent may have uncertainty about any of the parameters of the MDP including the transition function T and the reward function R. This uncertainty translates into a distribution over possible MDPs (all with the same state and action spaces in this paper). However the distribution over MDPs is represented, let the parameters of the distribution be denoted μ. Later we make precise the specific representation we use in this paper's experiments. The expected value for policy π over distribution μ is $E_{M\sim\mu}[V_M^\pi]$ where $M \sim \mu$ denotes a random MDP sampled from the distribution over the space of MDPs defined by parameters μ. The *Bayes-optimal policy* for distribution μ is $\pi_\mu^* = \arg\max_\pi(E_{M\sim\mu}[V_M^\pi])$, and the associated expected optimal value function for μ is therefore $E_{M\sim\mu}[V_M^{\pi_\mu^*}]$.

III. Expected Myopic Gain

We now consider what it would mean for an agent, whose current distribution over MDPs is parameterized by μ, to pose query q to the operator and receive response o. This will lead to a revised or posterior distribution over MDPs parameterized (by abuse of notation) as $\mu, < q, o >$. Our EMG algorithm computes an expected long-term gain in reward of asking q. The value can be compared with other possible queries, and with the costs of querying the operator, to decide which, if any, query to ask.

Gain: An agent in current state s_c with a distribution μ over possible MDPs expects value $E_{M\sim\mu}[V_M^{\pi_\mu^*}(s_c)]$ if it simply behaves forever according to π_μ^*. If the agent asks the operator query q and receives response o, its new Bayes-optimal policy would be $\pi_{\mu,<q,o>}^*$ and it should expect value $E_{M\sim\mu,<q,o>}[V_M^{\pi_{\mu,<q,o>}^*}(s_c)]$. How much did it gain by asking q and receiving answer o? One possibility is to define the gain as $E_{M\sim\mu,<q,o>}[V_M^{\pi_{\mu,<q,o>}^*}(s_c)] - E_{M\sim\mu}[V_M^{\pi_\mu^*}(s_c)]$ but this ignores one of the two effects of the new knowledge from the operator. This possible definition does account for the change in policy from π_μ^* to $\pi_{\mu,<q,o>}^*$ (the long-term exploitation effect) but ignores the change in the distribution over possible MDPs from μ to $\mu, < q, o >$ (the myopic exploration effect). To illustrate this point, suppose what it learns from the operator induces a distribution over MDPs where the expected value of the agent's current state is actually lower than before. In such cases, the gain as described above would be negative even though the agent would actually know more about its world (and would be able to use this information to change its policy). A better definition of gain should compare the value of the new optimal policy to the old optimal policy with respect to the *new* distribution over MDPs. Accordingly, we define gain as:

$$\begin{aligned} Gain(< q, o > |\mu, s_c) &= E_{M\sim\mu,<q,o>}[V_M^{\pi_{\mu,<q,o>}^*}(s_c)] \\ &\quad - E_{M\sim\mu,<q,o>}[V_M^{\pi_\mu^*}(s_c)]. \end{aligned} \tag{1}$$

This takes into account both effects of the new information from the operator, and moreover has the intuitively desirable property that $Gain(< q, o > |\mu, s_c) \geq 0$.

Since the agent does not know how the operator will respond to a query q, it predicts the probabilities of responses using its current distribution over MDPs defined by μ. Expected gain is thus (similar to Dearden *et al.* [2]):

$$E[Gain(q|\mu, s_c)] = \int Gain(< q, o > |\mu, s_c)\, P(o \,|\, \mu, q)\, do, \tag{2}$$

where we integrate over all possible operator responses o to query q.

EMG Query and Optimality: Assuming cost function C of querying the operator, our EMG algorithm asks query

$$(q^*|\mu, s_c) = \arg\max_{q\in Q} \Big(E[Gain(q|\mu, s_c)] - C(q)\Big), \tag{3}$$

provided $E[Gain(q^*|\mu, s_c)] > C(q^*)$, and where Q is the space of possible queries. Next we present a result that states the conditions under which EMG is optimal. Intuitively, the theorem below states that for the case of myopic exploration and long-term exploitation EMG is optimal.

Theorem 1. *For any distribution μ over MDPs and denoting the arbitrary current state of the agent as s_c, the EMG query as defined in Equation 3 is optimal under the following conditions:*

1) *The agent can ask exactly one query while in state s_c; and*
2) *The agent commits to a policy after receiving an answer to the one query in current state s_c and never changes that policy thereafter.*

Proof: (Sketch) The proof follows immediately from the definition of expected gain in Equation 2. Intuitively, the best the agent can do with respect to a distribution over MDPs if its knowledge of the distribution is fixed for all time is to execute the Bayes-optimal policy with respect to that distribution (this follows by definition of Bayes-optimal policy). Thus, given that μ defines the distribution

over MDPs for the agent and that the operator response will be determined by μ, the EMG query q^* is by construction the best the agent can do if it can only ask one query immediately and can never change its distribution over MDPs thereafter. ■

A. EMG Implementation

We have defined EMG as a general method for query selection, where query responses reveal some information about the agent's MDP distribution. A practical implementation must define a particular form for queries, and also be able to compute the effect query responses have on the MDP distribution. To avoid confounding factors in evaluating the core EMG algorithm, in this paper we focus on two types of queries for which computing posterior distributions is straightforward, and assume that the operator is capable of responding to those particular query types. In addition, since the agent's goal is to behave optimally with respect to the operator's model of the world, it *always* treats the operator's response as correct even if inconsistent with the agent's own experience of the environment. We now describe those query types, and summarize the computational strategies we chose to implement them.

Unknown Transition Probabilities: The agent knows the reward function of the operator's model of the true MDP exactly, but only has a prior distribution over a space of multinomial transition probabilities, expressed as independent Dirichlet distributions across the state-action space. That is, each state-action pair's distribution over next state is modeled as an independent Dirichlet distribution which amounts to maintaining a separate count of transitions to each possible next state (see [2] for more detail on modeling MDP distributions using the Dirichlet distribution). The form of the query q is an arbitrary state-action pair, i.e., $q = (s, a)$ and the form of the response is the transition probabilities for (s, a) from the operator's model. The update procedure overwrites the Dirichlet parameterization for (s, a) so that all MDPs sampled from the post-query distribution would have the transition probabilities for (s, a) set to the operator's response.

Unknown Reward Function: The agent knows the transition probabilities of the operator's model of the true MDP exactly but begins with a prior over a space of multinomial random reward functions, expressed as independent Dirichlet distributions across the state space (recall that in this paper we consider reward a function of state only). The form of the query q is a state s and the response is the multinomial reward distribution for s, which as before overwrites the Dirichlet representation.

Computational Challenges: Even for these simple query types, it is not clear that Equation 3 can be generally computed exactly due to three computational challenges: (1) the integral in Equation 2 is generally not analytically computable, (2) it is generally infeasible to compute the two Bayes-optimal policies for Equation 1, and (3) it is generally infeasible to compute the expected value, $E.[\cdot]$, of the Bayes-optimal policies for Equation 1. We handle (1) by approximating the integral using Monte-Carlo methods, sampling possible operator answers from the distribution $P(o \mid \mu, q)$ and averaging gain over the sampled operator answers. We handle (2) with the mean-MDP method below. Finally, we handle (3) again with Monte-Carlo methods, sampling MDPs from the appropriate distribution and averaging the values for the MDPs' Bayes-optimal policies. Obviously, for the two Monte-Carlo approximations, increased sampling improves approximation accuracy. For our experiments, we empirically determined that 200 (500) samples approximated transition (reward) queries well.

Mean-MDP method for computing Bayes-optimal policies: For the case of unknown reward functions, the Bayes-optimal policy is known to be the optimal policy with respect to the mean MDP [3]. For the case of unknown transition probabilities in acyclic domains, a similar result is easy to prove. In the more general case of cyclic domains, the mean-MDP method is only an approximation. However our empirical tests on small cases suggest it is generally a good approximation (henceforth, we refer to the optimal mean-MDP policy as Bayes-optimal even if approximate). An advantage of using independent Dirichlet distributions for both cases is that the mean-MDP is extremely simple to compute: one simply uses the counts to compute empirical probabilities for the mean-MDP parameters.

IV. Empirical Results

We examine EMG's performance on a pair of problem domains that are small enough to exhaustively compare EMG's choice against other possibilities, but that are interesting enough to allow us to begin testing the degree to which the conditions of Theorem 1 are critical for EMG's optimality. The first, called the "tree" domain, takes the form of a (acyclic) tree shown in Figure 1 (left), where the agent starts in state 0 and takes 2 actions, ending in one of the 4 "terminal" states $3-6$. In each of states 0, 1, and 2, the agent has two action choices, depicted with "solid" and "dotted" lines. For example, taking action s (solid) in state 0 has probability p_0^s and $1 - p_0^s$ of transitioning into states 1 and 2, respectively. The agent knows the transition topology (e.g., from state 0 the next state is either 1 or 2). For transition queries the agent also knows the expected rewards ($R4 = 0.4$ and $R5 = 1.0$ and zero for the other states). For reward queries, the agent also knows the transition probabilities (p_0^s = p_1^s = p_2^s = .9 and p_0^d = p_1^d = p_2^d = .8), that states $0-2$ have zero reward, and that rewards for states $3-6$ can be $\{0, 1, 2\}$.

Our second domain, called "grid" (Figure 1 right), has nine states, where the agent starts in state 0, and state 8 is the terminal goal state (the episode ends once the agent reaches it). The agent has a choice of four different actions

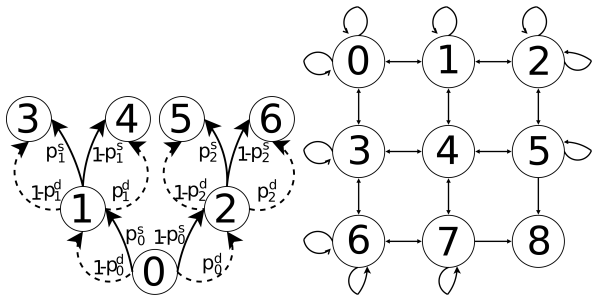

Figure 1. The "tree" and "grid" domains.

in each non-goal state, where each action can stochastically transition to neighboring states as indicated by the solid arrows. The agent knows this topology. For transition queries, the agent also knows the rewards of all states ($R5 = -10$, $R6 = -2$, $R8 = 1$, and the rest are all 0). For reward queries, the agent also knows the actions' transition probabilities: each action (conceptually, North, South, East, and West) moves in its corresponding direction with probability .9 and in the opposite direction with probability .1, and knows that rewards can be $\{-10, -2, 0, 1\}$ for all states.

As described in the previous section, for both query types in both domains, the agent's prior is a set of independent Dirichlet distributions. All counts are initialized to 1 so that the prior is uniform over multinomial transition (reward) probabilities for transition (reward) queries. Also, in all of our experiments, the operator had a correct model of the environment.

A. Experimental Procedure

For a particular type of query and a particular domain, the agent computes the Bayes-optimal policy given its initial knowledge, and it also computes the EMG query. To evaluate the actual *value* of asking a query, we adopt the following procedure. We randomly draw some number (10,000 unless otherwise noted) of sample (fully specified) MDPs from the prior distribution. For a given sampled MDP and a query, we use the sampled MDP's answer to the query to update the agent's distribution over MDPs, from which it computes a (possibly new) Bayes-optimal policy. The query's value for the MDP is the difference in long-term expected reward from executing the new versus the old policy in the sampled MDP.

Given such a measure for the value of each query, we can compare EMG to other query selection methods by studying the average value of the query selected by each method across the sample MDP set. The first method we analyze is *random*, which simply chooses a query randomly. We execute this strategy by choosing a query randomly for *each* sample MDP (as opposed to using the same query throughout the sample MDP set). The next query selection method, *best-fixed*, asks the query with maximum average value across the sample MDP set. The last method, *omni*scent, chooses the query with highest value separately for each sample MDP (so it may choose a different query for each sample). Of course, neither Best-Fixed nor Omni are realizable query selection methods, as they both identify queries to ask using after-the-fact information that the agent could not possibly have. We include them in order to get a sense for how well EMG performs relative to upper bounds (best-fixed is an upper bound since both it and EMG must select 1 query for the entire sample set, and best-fixed chooses the query with best average value; omni is an upper bound since it selects the best query for each individual sample, resulting in a set of optimal queries across the sample set).

As an example, suppose we are considering reward queries in the tree domain and draw 2 MDPs as our sample set. Random will choose a random state to ask about for each sample; suppose it chooses states 0 and 1, respectively. EMG will choose to ask about state 3 (as explained in the next section). Best-fixed will look at the two sample MDPs in the set and choose to ask about whatever state yields the highest average value across both sample MDPs; suppose that is state 4. Omni will choose the best state to ask about for each sample; suppose it chooses states 1 and 2, respectively. Our evaluation of each query selection method then consists of averaging the value of each method's queries throughout the sample set: in this example, random's choice of (0,1), EMG's choice of (3,3), best-fixed's choice of (4,4), and omni's choice of (1,2).

Due to symmetries in our environments, an arbitrary choice sometimes is made over equally good (in expectation) queries, and what we see experimentally is that due to the luck of the draw sometimes the EMG choice is the same as the best-fixed query for the specific trials, and sometimes it is not. To account for this variability, our data presents results that average the value of each query selection method over 10 independently-generated experiments of 10000 sampled MDPs each.

B. EMG Optimality

We begin by comparing in Table I the average values of EMG, best-fixed, omni, and random queries in the tree and grid domains. For transition queries in the tree domain, the EMG query is arbitrarily either p_2^s or p_2^d. Though derivable analytically, we can intuitively explain why asking about an action in state 2 is rational. State 2 dominates state 1 because (since they are equally likely to be reached) a better decision in state 2 in expectation will be more beneficial given the high reward of state 5. Comparing states 0 and 2, information about an action in either would be expected to equally increase the probability of reaching state 5. However, such a better decision in state 0 also *decreases* the probability of reaching state 4 (the only other positive reward), while a better decision in state 2 leaves the probability of reaching state 4 unchanged. Hence, asking about a state 2 action is

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree	0.0137	0.0202	0.0203	0.0473
Trans	Grid	0.382	0.715	0.740	3.12
Reward	Tree	0.0530	0.0576	0.0586	0.127
Reward	Grid	0.851	1.23	1.34	4.04

Table I
RESULTS FOR THE OPTIMALITY EXPERIMENT.

the optimal choice (in expectation). The top row in Table I shows that EMG performed considerably better than random, and approaches best-fixed as we would hope. Omni provides an upper bound for how well the agent could do if it could miraculously guess the right query for every sample MDP.

In the grid domain, the EMG transition query asks about an action in state 4 (intuitively, the point on the best path to goal state 8 where a misstep is most costly). With 4 actions in state 4 that are initially indistinguishable, its chances of guessing the best for a particular set of trials is smaller than in the tree domain, so EMG is more noticeably lower than best-fixed. The optimal reward query in the tree domain is to ask about state 3, since the agent has the most control in getting to state 3 if it finds out it has high reward, and also the most control in avoiding state 3 if it finds out it has low reward. For reward queries in the grid domain, the symmetry of transition probabilities means that asking about states 1 and 3 are equivalently the best choice, since the agent must pass through at least one of them on its path to the goal and it may choose which to pass through. Table I shows that in the grid domain, EMG's performance beats that of random and approaches best-fixed, just as in the tree domain.

Accounting for cost: The above confirms empirically that EMG finds optimal queries when its conditions are met, but recall that even the optimal query should only be made when its expected gain is no less than its cost (Equation 3). The value of each entry in Table I provides an empirically-derived maximum cost the agent should be willing to incur for posing that query; otherwise in expectation it would do better by autonomously following its initial Bayes-optimal policy than asking for help. We confirmed that EMG predicts the value of its query well by comparing the expected gain computed by EMG for the optimal query with the corresponding empirical gain in Table I. For example, 100 independent executions of our implementation of EMG for transition queries on the tree domain yielded an average expected gain of 0.0212 (st.dev. 0.0018), which closely estimated the corresponding empirical gain in Table I, 0.0202. But we next examine how EMG's predictions about query value and its willingness to query can change as its optimality conditions are violated.

C. Post-Query Updates

The empirical results so far have served largely to illustrate and benchmark what we already justified theoretically: when the myopic assumptions are met, EMG will find a query that, in expectation, will be optimal. Throughout the remainder of our experiments, we develop insights into how well EMG can perform even when the myopic assumptions are violated.

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree-1	0.0242	0.0298	0.0299	0.0530
Trans	Grid-1	0.739	1.03	1.05	3.18
Trans	Tree-2	0.0336	0.0382	0.0385	0.0566
Trans	Grid-2	1.05	1.35	1.35	3.35
Reward	Tree-1	0.0796	0.0819	0.0825	0.129
Reward	Grid-1	1.46	1.82	1.84	4.04
Reward	Tree-2	0.102	0.103	0.104	0.131
Reward	Grid-2	1.97	2.27	2.28	4.08

Table II
RESULTS WITH 1 AND 2 POST-QUERY UPDATES.

EMG's optimality conditions in Theorem 1 require that once the agent receives the operator's response to its query (if it chooses to ask) and builds its new Bayes-optimal policy, it cannot update its policy any further. For example, an agent executing in a state space that includes cycles (like the grid domain) could potentially learn from its experiences to update its model so as to build a better policy given it might return to that state. The EMG algorithm's myopic exploration does not consider how a state-action pair that can be learned at runtime might be less valuable to ask about beforehand.

Because of the myriad ways in which an agent could gather and incorporate knowledge at runtime, we examine the same phenomenon but in a simpler way. Our experiments assume that *after* it asks its one query, an agent is also unexpectedly (to it) given information for either 1 or 2 randomly-selected queries, where a random query could duplicate what the agent asked, and in experiments with 2 random queries they are distinct. The entries in the table represent the average value for knowing both the answer to its query *and the post-query information*.

As shown in Table II, EMG continues to track best-fixed closely, suggesting that EMG's initial query choice (which is the same as for the previous set of experiments) tends to remain optimal despite not anticipating the possible additional information. Obviously, it would not be difficult to construct other problems where EMG performs much more poorly, such as problems where it is known up-front that the same information that EMG asks for will definitely also be in the post-query information; being myopic, EMG cannot use this knowledge, whereas a non-myopic approach could.

Notice also in Table II the narrowing gap between EMG/best-fixed and random, and with omni. This is not surprising, since given more randomly-chosen additional information, we would expect the agent's initial query choice to make less difference. This has implications on the value of incurring the overhead of EMG (random might become preferable) as well as on whether to query at all.

Accounting for cost: As the impact of the EMG query is diluted by other gathered information, the value-added

by the query will fall. For example, we've experimentally determined that the value-added of the EMG query in row 1 Table II (by comparing to the case where no query is asked but random information is received) is 0.0164, and for row 3 is 0.0121. EMG, as we have seen, computes a gain of 0.0212 for its query, and thus EMG could myopically overestimate the acceptable cost for querying the operator.

D. Sequences of Queries

EMG also assumes that the agent can only ask a single query. If the agent can ask multiple queries, then EMG can make incorrect decisions by failing to factor into its gain calculations the (exploration) value a query has for asking better queries downstream. In general, computing optimal sequences of queries to reliably derive such a strategy involves solving Bayes-Adaptive Markov decision processes which are known to be intractable [4].

The hypothesis we will empirically test is as follows: If an agent can ask two queries, then using EMG to determine these queries yields an expected added value competitive with the best possible query sequence. We tested this hypothesis in our domains by slightly modifying our experimental procedure: for each trial (sampled true MDP), after computing and asking the EMG query and incorporating the response, the agent ran EMG a second time to identify the most valuable (next) query to ask given its new distribution over MDPs. After the response to this, the Bayes-optimal policy given the combined information of the two responses was evaluated against the true MDP. For each of the 10,000 trials, we also computed the value for every possible fixed *pair* of queries.

As seen in Table III, the sequence of two EMG queries together yielded an average value similar to that of the average value of the best-fixed-pair (omni and random query-pair values are also given for completeness). In fact, in some cases, EMG did even better. EMG has an advantage over any fixed-pair, because EMG is *not* fixed: it can choose a different second query based on the response to the first. On the other hand, EMG is at a disadvantage because it does not look ahead to how the first query can "set up" the second.[1] As the data shows, sometimes EMG's advantage of following a policy (conditionally branching to a different second query) outweighs being myopic, and other times it does not. Unfortunately, experimentally evaluating the continuous space of all possible 2-step policies to see how close EMG gets to the best such policy is infeasible.

Accounting for cost: We can of course ask even longer sequences of queries. Figure 2a shows, for the transition-query grid domain, a comparison between the value of the posterior policies induced by sequential EMG or random queries, as well as the optimal (in expectation) policy the agent is attempting to obtain. We see that repeatedly asking EMG queries improves agent value faster than a sequence of random queries, which is not very surprising. The important thing to note, however, is in general the improvement in value for each additional query decreases as the sequence gets longer. This suggests that, if EMG is used for problems where multiple (even unlimited) queries can be asked, there will likely be a tipping point where the cost of querying exceeds the expected gain of doing so. This is shown in Figure 2b, where a few different costs for querying are shown. The higher the cost, the fewer queries can be asked before the expected net value to the agent starts falling.

[1]If each query EMG considered was instead about a pair of pieces of information, EMG could find the pair with the highest expected gain, but then would be limited to considering a fixed pair.

Query	Domain	Rand	EMG	BF	Omni
Trans	Tree	0.0254	0.0386	0.0379	0.0592
Trans	Grid	0.640	1.17	1.25	3.98
Reward	Tree	0.0907	0.0960	0.0968	0.134
Reward	Grid	1.60	2.31	2.13	4.42

Table III
RESULTS FOR THE SEQUENCE OF QUERIES EXPERIMENT.

E. Online Query Asking

Finally, we try relaxing Theorem 1's condition that the agent ask its query before taking any action. Now, an agent could benefit by waiting to ask its query until after taking some actions. In the tree domain for transition queries, for example, as we have seen if the agent must ask its query before it begins acting, it should ask about an action in state 2. However, the response will not change the agent's action choice in state 0. If it takes its action in state 0 first, taking it to state 1 or 2 with equal probability in expectation, then it can ask about an action in whichever of those states it ended up in. Its expected value increases because it improves the chances of reaching a non-zero reward state however its action in state 0 turns out, compared to asking first (where information it gets about an action in state 2 will only be useful 50% of the time in expectation).

This suggests a heuristic strategy for deciding when to query: if the Bayes-optimal policies for every possible answer to the (proposed) EMG query all prescribe the same action for the current state, there is no advantage in asking the query at this time. The agent should instead take an action. It can then repeat this procedure, postponing asking a query until it has reached a state where that query will impact its next action choice.

This Ask When Impacts Next Action (AWINA) heuristic clearly would work in the simple example we've been considering so far. However, because the tree domain involves taking exactly two actions, it does not provide much latitude for more comprehensive evaluation, so we use the grid world (with all other EMG optimality conditions met) to compare standard (prior to any actions) EMG to the AWINA heuristic. We also consider the case where the decision about asking the EMG query is made randomly (with probability .5) at each time step (until a query has been asked), and the results

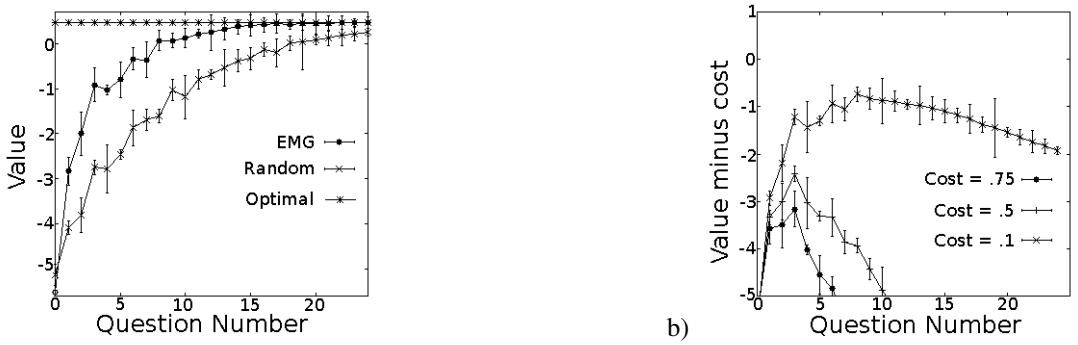

Figure 2. a) Value of the policy obtained after a sequence of EMG queries, compared with a random query sequence and the optimal policy with full information. b) Value of the policy obtained after a sequence of EMG queries, minus the total cost to obtain it for various costs of querying.

Query	EMG	EMG+AWINA	EMG+Random
Trans	0.715	0.869	0.750
Reward	1.23	1.23	1.18

Table IV
RESULTS FOR THE ONLINE EXPERIMENTS (GRID DOMAIN).

are given in Table 4. These results suggest that, in the grid domain, the agent will do well to use EMG with the AWINA heuristic when dealing with transition queries because it is better off waiting until it knows which negative state it has wandered near. Note that even waiting randomly to ask does better than always asking at the start. For reward queries, in contrast, finding out anything about the reward landscape before setting off is useful, so AWINA does not improve EMG, and EMG+random does worse than the others. In summary, AWINA can sometimes help, but a more complete exploration for additional interesting heuristics is an area for future research.

Accounting for Cost: The costs of querying induce another simple heuristic: postpone querying until the expected gain exceeds the querying cost by some amount. For example, in the tree domain with transition queries discussed earlier, the EMG query's expected gain when the agent is in state 0 is lower than the gain for that same query when it is in state 2. Similarly, the gains for queries about state 1 dominate after the agent moves from state 0 to 1. If the operator cost is greater than the expected gain for asking while in state 0 but less than that of asking while in states 1 or 2, the agent will correctly delay asking its one query. Thus, the EMG computation in Equation 3 implicitly helps balance the value of asking about a distant state far enough in advance to improve near-term decisions, against the disutility of getting information that environmental stochasticity might render immaterial.

V. SCALING TO LARGER PROBLEMS

In this paper, we aimed to introduce EMG as a general query strategy for agents in MDPs. As such, our evaluation of EMG considered toy problems, chosen to be small enough to allow exhaustive evaluation of EMG's query choices but flexible enough to allow for testing the violation of EMG's optimality conditions. Applying EMG to larger, more realistic problems inevitably involves computation reduction afforded by particular query types or induced by heuristics.

A. Other Query Types

A strength of EMG is its generality in the form queries may take. However, this generality is a large factor in the intractability of Equation 2. Our reward and transition query implementations assumed that query answers may span a continuous interval, resulting in the necessity for Monte-Carlo methods to evaluate Equation 2. However, for queries whose possible responses span a small discrete set, using EMG becomes significantly more practical as the integral in Equation 2 becomes a sum with relatively few terms. As an example, for reward queries, if the agent knows that reward values may only take on the values 0, 1, or 2, Equation 2 can be computed exactly rather than relying on Monte-Carlo methods.

A particularly powerful application of queries is when there is special structure in the domain. Namely, if the domain may be represented compactly as a parameterization depending on some property of the MDP, the agent may query those properties and infer a great deal of information about its world while only needing to consider a small number of queries, even if the world is quite large. Our preliminary experiments running EMG on a domain with 10000 states but a compact parameterization have so far been positive.

B. Applying Heuristics

There exist a variety of heuristics one can apply in order to trade computational efficiency for further approximation of the EMG calculation. Since EMG's computation scales linearly with the number of queries it considers, a natural inclination is to prune the set of queries considered. As an example, for the reward and transition query types discussed in this paper, queries depending on nearby states are most frequently chosen by EMG, so a simple heuristic would be to consider only the K nearest states as queries,

where "nearest" and K are determined depending on the application. Another example would be to only consider queries that disambiguate what the agent should do at key parts of the statespace, such as choke points or intersections, where the choice of action greatly influences the portion of the statespace the agent spends time in. For a given domain and query type, other heuristics that only consider a subset of possible queries might make sense.

VI. Other Selective Querying Approaches

Other selective querying approaches in the literature generally focus on queries that are demonstration requests ("What is an optimal action for state s?"). Chernova and Veloso [5] describe an interactive policy learning approach, which begins with an offline learning from demonstration [6] phase where the agent estimates a policy from demonstration data using supervised learning. Their approach then transitions to an online phase where the agent can request a demonstration for its current state if it cannot *confidently* classify (choose an action for) that state. The method relies on the assumption that the same or similar actions are optimal in similar regions of the state space, rather than on learning to improve an MDP model over which it could perform sequential reasoning. Without a model, the method cannot make predictions about downstream effects of actions. For example, suppose any action taken in a given state will take the agent to the same next state s – this implies that any action for s is optimal. But the agent only has a policy and confidence values for the policy in each state, and if it has low confidence for s it may ask for help. As such, the method proposed in [5] can sometimes ask unnecessary queries that a model-building agent, such as one using EMG, could filter out (EMG would not output a positive gain for such a query, since any response from the operator would not cause the agent to change its policy).

Active learning [7] is another concept similar to ours, but differs in that it involves querying in an offline manner. Nevertheless, strategies for selecting training data for active learning could be applied in an online setting as well. In particular, recent work by Lopes *et al* [8] presents an active learning method for inverse reinforcement learning [9], where potential demonstration requests are ranked based on an entropy measure of the optimal action density (estimated from acquired demonstration data and prior knowledge) for their corresponding states. However, in contrast to EMG, their query ranking algorithm does not take into account the agent's current state and as a result their algorithm may query states that the agent is unlikely to ever reach.

VII. Conclusions and Ongoing Work

We have introduced the EMG algorithm and identified conditions under which it computes an optimal single query. By comparing the expected gain of the optimal query to the cost of asking that query, an agent can determine whether to ask for help, or to autonomously pursue its current best policy. Our empirical results confirm EMG's optimality when its myopic conditions are met, and that it can often continue to be effective even when the conditions are violated.

Our ongoing work is examining other query types, such as demonstration requests: "What would you do in state s" where s can be a hypothetical future state. The operator's response indirectly provides information about her model of transitions and rewards for inducing a posterior distribution over the MDPs. To estimate this distribution, we have been drawing on ideas from Bayesian Inverse Reinforcement Learning [3], as well as particle-filtering techniques. In addition, we have been investigating methods to scale EMG to larger and more realistic problems.

Acknowledgments: This work has been supported in part by the Ground Robotics Reliability Center of the University of Michigan, a Center of Excellence sponsored by the U.S. Army TARDEC. The authors thank Michael Wellman and David Karmol for contributing to these ideas.

References

[1] Crandall and Goodrich, "Experiments in adjustable autonomy," *Systems, Man, and Cybernetics, 2001 IEEE Int Conf on*, vol. 3, pp. 1624–1629 vol.3, 2001.

[2] R. Dearden, N. Friedman, and D. Andre, "Model based bayesian exploration," in *UAI*, K. B. Laskey and H. Prade, Eds. Morgan Kaufmann, 1999, pp. 150–159.

[3] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *IJCAI*, M. M. Veloso, Ed., 2007, pp. 2586–2591.

[4] M. O. Duff, "Design for an optimal probe," in *ICML*, T. Fawcett and N. Mishra, Eds., 2003, pp. 131–138.

[5] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *J. Artif. Int. Res.*, vol. 34, no. 1, pp. 1–25, 2009.

[6] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[7] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Mach. Learn.*, vol. 15, no. 2, pp. 201–221, 1994.

[8] M. Lopes, F. Melo, and L. Montesano, "Active learning for reward estimation in inverse reinforcement learning," in *ECML PKDD*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 31–46.

[9] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *ICML*, P. Langley, Ed. Morgan Kaufmann, 2000, pp. 663–670.