

Sampling-based Motion Planning for Robotic Information Gathering

Geoffrey A. Hollinger

School of Mechanical, Industrial & Manufacturing Engineering
Oregon State University
Corvallis, OR 97331, USA
Email: geoff.hollinger@oregonstate.edu

Gaurav S. Sukhatme

Department of Computer Science
University of Southern California
Los Angeles, CA 90089, USA
Email: gaurav@usc.edu

Abstract—We propose an incremental sampling-based motion planning algorithm that generates maximally informative trajectories for guiding mobile robots to observe their environment. The goal is to find a trajectory that maximizes an information metric (e.g., variance reduction, information gain, or mutual information) and also falls within a pre-specified budget constraint (e.g., fuel, energy, or time). Prior algorithms have employed combinatorial optimization techniques to solve these problems, but existing techniques are typically restricted to discrete domains and often scale poorly in the size of the problem. Our proposed Rapidly-exploring Information Gathering (RIG) algorithm extends ideas from Rapidly-exploring Random Graphs (RRGs) and combines them with branch and bound techniques to achieve efficient optimization of information gathering while also allowing for operation in continuous space with motion constraints. We provide a rigorous analysis of the asymptotic optimality of our approach, and we present several conservative pruning strategies for modular, submodular, and time-varying information objectives. We demonstrate that our proposed approach finds optimal solutions more quickly than existing combinatorial solvers, and we provide a proof-of-concept field implementation on an autonomous surface vehicle performing a wireless signal strength monitoring task in a lake.

I. INTRODUCTION

Mobile robots are increasingly being tasked with gathering information about their environment. Emerging application domains include marine monitoring [21], aerial surveillance/search [8], tactile object recognition [10], and facility inspection [9]. In all of these domains, the goal is to maximize some metric of information (e.g., quality of the survey, probability of locating a target, or accuracy of the inspection) while satisfying constraints on fuel, energy, or time.

The *informative motion planning* problem of maximizing information gathered subject to a budget constraint is particularly challenging because it typically requires searching over a large and complex space of possible trajectories. Such problems have been shown to be NP-hard [20] or even PSPACE-hard [18] depending on the form of the objective function and the space of possible trajectories. Prior work has leveraged approximation algorithms [20] and branch and bound solvers [2] to provide informative trajectories for mobile robots. However, these prior algorithms are limited to discrete domains and often scale poorly in the amount of budget and the size of the environment.

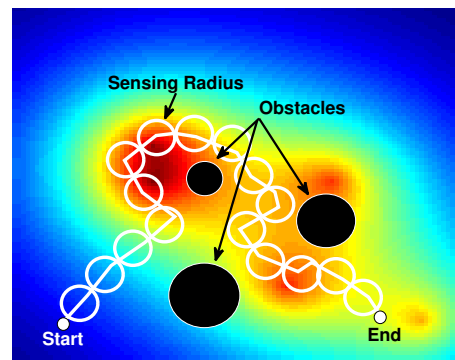


Fig. 1. Example information landscape and trajectory generated by the proposed algorithm for an autonomous vehicle monitoring a phenomenon of interest (e.g., an ocean event, seismic activity, or an area to be surveyed). The red areas have higher information quality than the blue areas. Our algorithm extends ideas from stochastic motion planning to provide informative trajectories that satisfy a budget constraint.

In this paper, we provide an algorithm that solves the informative motion planning problem using iterative sampling. Our approach combines ideas from Rapidly-exploring Random Graphs (RRGs) [12] with insights from branch and bound optimization [2] to provide improved efficiency and generality. Adapting theoretical analysis from the RRT* algorithm also allows us to show asymptotic optimality for a large class of objective functions. Prior algorithms, including Information-rich RRTs [16] and Belief-space RRTs [4], have been shown to solve related problems for certain objective functions. The key novelty of this paper is the introduction of an incremental sampling-based algorithm for generating informative trajectories that is asymptotically optimal, outperforms existing combinatorial solvers, and allows for general motion constraints.

The remainder of this paper is organized as follows. We first discuss related work to highlight the need for efficient informative motion planning algorithms (Section II). We then provide a formalization of the informative motion planning problem (Section III), and we introduce the proposed sampling-based algorithm along with a rigorous analysis of the asymptotic optimality of our approach (Section IV). To validate our algorithm, we compare it to existing combinatorial solvers, and we also provide a field implementation on an autonomous

surface vehicle (Section V). Finally, we draw conclusions and discuss avenues for future work (Section VI).

II. RELATED WORK

Autonomous information gathering problems have a rich history in mathematics, computer science, and robotics dating back to early work in sequential hypothesis testing [23]. Early research was concerned with the problem of determining which experiments to run to best determine the nature of an unknown. Similar ideas were then generalized to account for constraints on physical systems, which resulted in a vibrant community studying active perception [1].

Active perception problems were later extended to account for mobile sensing within the framework of Bayesian reasoning [5]. Subsequent works have resulted in an extensive suite of solutions that incorporate information theoretic measures for such problems as object recognition, mapping, and scene reconstruction (see [6] for a survey). Modern applications of next best view planning and belief space planning continue to push the envelope of gradient-based optimization in these domains [11]. While such algorithms have been shown to be useful for a range of domains, they typically rely on restrictive assumptions on the objective function and do not have guarantees on global optimality. Finite-horizon model predictive control methods [3, 19] provide improvement over myopic techniques but also do not have performance guarantees beyond the horizon depth.

A number of general solvers for robotic information gathering problems utilize combinatorial optimization techniques to search over a discrete grid. The recursive greedy algorithm [20] is one example that achieves bounded performance for submodular objective functions. Branch and bound techniques have also been proposed that only require the objective function to be monotonic [2]. The resulting solvers typically require computation exponential in the size of the problem instance due to the large blowup in the search space with increasing budget. An alternative is to utilize a finite-horizon solver that solves the problem for only a portion of the budget at a time [8]. While such solvers perform heuristically well, they do not carry guarantees for behavior outside the horizon length.

The use of sampling-based motion planning algorithms, such as the Rapidly-exploring Random Tree (RRT) [15] and probabilistic roadmap (PRM) [14], has increased enormously in past years. Such algorithms have the advantage of quickly exploring a space of interest to achieve a feasible solution. The development of the RRT* and PRM* algorithms has led to algorithms that asymptotically approach the optimal solution with increasing computation time [12]. Variants have also been proposed that have anytime capabilities and utilize branch and bound techniques to trim the search tree [13]. Recent work has shown that extensions of these sampling-based algorithms can solve problems with uncertainty in position while preserving asymptotic optimality guarantees [4]. These algorithms are concerned solely with minimizing a cost function, and they do not consider the problem of maximizing an information metric.

As such, they cannot directly be used to solve informative motion planning problems.

Sampling-based algorithms are natural candidates for generating motion plans for information gathering tasks. The Information-rich RRT (iRRT) was designed to maximize the accuracy of tracking a mobile target [16]. The iRRT extends sampling-based algorithms to solve a class of information gathering problems. However, the authors do not provide asymptotic optimality guarantees, and the application is limited to tracking problems. In the current paper, we introduce the Rapidly-exploring Information Gathering (RIG) algorithm that combines ideas from the iRRT [16], RRT* [12], RRBT [4], and combinatorial branch and bound [2] algorithms to provide efficient asymptotically optimal information gathering for a rich space of objective functions.

III. PROBLEM SETUP

In the general case, informative motion planning requires solving the following maximization problem:

$$\mathcal{P}^* = \operatorname{argmax}_{\mathcal{P} \in \Psi} I(\mathcal{P}) \text{ s.t. } c(\mathcal{P}) \leq B, \quad (1)$$

where Ψ is the space of possible trajectories for a robot or team of robots, B is a budget (e.g., time, fuel, or energy), and $I(\mathcal{P})$ is a function representing the *information* gathered along the trajectory \mathcal{P} .¹ We will assume the robots are modeled using discrete time dynamics, and that the trajectory is deterministic given the environment and the control inputs. As a notational convention, we will denote the portion of a trajectory from times t_0 to t_f as $\mathcal{P}^{t_0:t_f}$. Where it will not cause confusion, we will also denote a partial trajectory \mathcal{P}^t as the segment of a trajectory centered around time t . We allow for restrictions on the space of valid trajectories, including kinodynamic constraints, obstacles, and vehicle speed limitations.

We assume in this paper that the form of the cost and information objectives are known a priori. Regarding the cost function, we assume it is strictly positive, monotonically increasing, bounded, and additive. These assumptions include most objective functions utilized in prior sampling-based motion planning literature (e.g., distance, energy, etc.) [12]. Note that these assumptions ensure that all trajectories satisfying the budget will be of finite length.

We consider information objective functions of the following three types: modular, time-varying modular, and submodular. If we let \mathcal{P}_a and \mathcal{P}_b be two partial trajectories, and we let \mathcal{P}_{ab} be the trajectory found by combining them, we define a modular information objective as one where $I(\mathcal{P}_{ab}) = I(\mathcal{P}_a) + I(\mathcal{P}_b)$. A time-varying modular objective is the same, except that the value of $I(\mathcal{P}_{ab})$ depends on the time when \mathcal{P}_a and \mathcal{P}_b are executed. In contrast, a submodular objective is one where $I(\mathcal{P}_{ab}) \leq I(\mathcal{P}_a) + I(\mathcal{P}_b)$. In the context of robot motion planning, one key property of submodular

¹We note that RIG can potentially be used to optimize any quantifiable metric of interest along the trajectory. The scope of this paper is limited to information gathering objectives, but the examination of additional objectives of a similar form is an interesting avenue for future work.

objectives is that the amount of information gathered in the future is dependent on the prior trajectory, whereas with modular objectives it is not.

Two major factors make solving these optimization problems particularly difficult: (1) for nearly all interesting objective functions, finding the optimal solution is formally hard (NP-hard or PSPACE-hard) [18, 20], and (2) the space of trajectories Ψ grows with increasing budget, making exhaustive searches intractable. We propose an incremental algorithm that utilizes sampling to generate increasingly informative trajectories that satisfy the cost budget constraints. This approach allows for the generation of informative trajectories with complex information objectives.

IV. ALGORITHM DESCRIPTION

We now discuss the proposed motion planning algorithm that efficiently generates trajectories to maximize an information metric while also maintaining budget constraints. The key idea is to sample the configuration space of the vehicle and to build up a graph of possible trajectories by incrementally extending candidate trajectories towards the sampled points.

The full Rapidly-exploring Information Gathering (RIG) algorithm is described in Algorithm 1. The main loop of the algorithm begins with a start node and generates randomly sampled points in the configuration space. For each new point that is sampled, the algorithm extends all nearby nodes towards the newly sampled point to generate a graph of solutions. Nodes in the graph are only extended if the resulting cost does not exceed the budget, and nodes that are not promising can be pruned (see Section IV-C).

After nodes are extended, the cost and information at each new node is calculated, and an additional step is required where the information and cost is recursively updated along all the new edges. During this recursive update, new nodes are generated to store the information and cost generated by the trajectories formed from the newly added edges. More nodes are then generated at the neighbors of these newly added nodes and so on until all eligible nodes have been expanded. Pruning can also be employed during this step to limit the number of nodes added. After the algorithm has run for many iterations, the graph will contain a number of possible informative trajectories within the budget constraint.

The general RIG algorithm allows for constraints on the vehicle’s motion through the use of a Steer() function to extend nodes towards newly sampled locations. Requirements on the steer function are discussed in the next section. Points on the graph are extended using a Near() function. The near function can be heuristically set or set based on pre-specified ball around the sampled point (as in [12]). The algorithm can also account for obstacles in the environment by limiting feasible trajectories to a free configuration space \mathcal{X}_{free} .

The main challenges presented by the information gathering problem are (1) focusing the graph generation such that candidate trajectories satisfy the budget requirements, (2) managing computational complexity for computing the information gathered at each node on the graph, and (3) pruning

Algorithm 1 Rapidly-exploring Information Gathering (RIG)

```

1: Input: Step size  $\Delta$ , Budget  $B$ , Workspace  $\mathcal{X}_{all}$ ,
   Free space  $\mathcal{X}_{free}$ , Environment  $\mathcal{E}$ , Start config.  $\mathbf{x}_{start}$ 
2: % Initialize cost, information, and starting node
3:  $I_{init} \leftarrow InitialInformation(\mathbf{x}_{start}, \mathcal{E})$ 
4:  $C_{init} \leftarrow 0, n \leftarrow \langle \mathbf{x}_{start}, C_{init}, I_{init} \rangle$ 
5: % Initialize vertex list, edge list, and graph
6:  $V \leftarrow \{n\}, V_{closed} \leftarrow \emptyset, E \leftarrow \emptyset, \mathcal{G} \leftarrow (V, E)$ 
7: while processing time remains do
8:   % Sample configuration space of vehicle
9:    $\mathbf{x}_{samp} \leftarrow Sample(\mathcal{X}_{all})$ 
10:  % Find near points to be extended
11:   $N_{near} \leftarrow Near(\mathbf{x}_{samp}, V \setminus V_{closed})$ 
12:  for all  $n_{near} \in N_{near}$  do
13:    % Extend towards new point
14:     $\mathbf{x}_{new} \leftarrow Steer(\mathbf{x}_{n_{near}}, \mathbf{x}_{samp}, \Delta)$ 
15:    if  $NoCollision(\mathbf{x}_{n_{near}}, \mathbf{x}_{new}, \mathcal{X}_{free})$  then
16:      % Calculate new information and cost
17:       $I_{new} \leftarrow Information(I_{n_{near}}, \mathbf{x}_{new}, \mathcal{E})$ 
18:       $c(\mathbf{x}_{new}) \leftarrow EvaluateCost(\mathbf{x}_{n_{near}}, \mathbf{x}_{new})$ 
19:       $C_{new} \leftarrow C_{n_{near}} + c(\mathbf{x}_{new})$ 
20:       $n_{new} \leftarrow \langle \mathbf{x}_{new}, C_{new}, I_{new} \rangle$ 
21:      if PRUNE( $n_{new}$ ) then
22:        Delete  $n_{new}$ 
23:      else
24:        % Add edges and vertex
25:         $E \leftarrow E \cup \{(n_{new}, n_{near}), (n_{near}, n_{new})\}$ 
26:         $V \leftarrow V \cup \{n_{new}\}$ 
27:        % Add to closed list if budget exceeded
28:        if  $C_{new} > B$  then
29:           $V_{closed} \leftarrow V_{closed} \cup \{n_{new}\}$ 
30:        end if
31:         $\mathcal{G} \leftarrow (V, E)$ 
32:        % Recursively add nodes along new edges
33:         $\mathcal{G} \leftarrow UpdateInfoAndCost(n_{new}, \mathcal{G})$ 
34:      end if
35:    end if
36:  end for
37: end while
38:  $\mathcal{P} \leftarrow MaxInformationPlan(\mathcal{G})$ 

```

partial trajectories that cannot lead to a more informative final trajectory than the current best. We now discuss how we can overcome these challenges.

A. Budgeted Trajectory Generation

The problem of maximizing information subject to a budget constraint differs in several ways from problems typically solved using sampling-based motion planners. In many problem domains, a vehicle must move from one point to another while minimizing the trajectory cost [12]. For the problem of information optimization, there is no fixed goal point. Instead, there is a hard constraint on budget at which point the mission

time has expired or the vehicle has run out of fuel.²

To apply sampling-based motion planners to these problems, we make the following modification: if a candidate trajectory would exceed the budget, it will never be extended towards a sampled point. We maintain a *closed list* of nodes on the graph that represent completed trajectories. The proposed method can also be used in a receding-horizon manner by planning over a budget increment and then re-planning after the budget increment is expended. As new points are generated, trajectories that are not completed will be extended and eventually lead to completed trajectories. This approach builds up a large number of completed trajectories that efficiently explore the space of trajectories and provide different levels of information maximization.

B. Computational Complexity

As described above, we assume that given a partial trajectory $\mathcal{P}^{t_0:t_f}$ for times t_0 to t_f , we can calculate the information gathered by some known function $I(\mathcal{P}^{t_0:t_f})$. For modular and time-varying modular functions, it is straightforward to build trajectories in an incremental fashion by storing the information gathered at $t-1$ and then adding the incremental information gathered at t . For submodular functions, the entire partial trajectory is necessary to calculate the information at time t . Thus, the trajectory must be reconstructed by traversing the graph backwards, which is an $O(N)$ operation.

For some objective functions, the cost of calculating the information gathered may also increase as the length of the trajectory grows. The computational requirement for adding a new node grows as $O(N + f(N))$, where $f(N)$ is the computational cost of calculating $I(\cdot)$ for a trajectory of length N . The locations of the active nodes are stored in a KD-tree, which allows for efficient retrieval of the near nodes. The memory requirements grow linearly in the number of nodes since the stored information remains constant.

It is important to note that these computational requirements grow in the number of nodes added to the graph. Since all of the near nodes are extended for each new sample, denser graphs will add more nodes per iteration than sparser graphs. This additional computation provides motivation for pruning away trajectories that are no longer useful for finding the best information gathering trajectory. We will next discuss how to develop pruning strategies using formal analysis of asymptotic optimality.

C. Theoretical Analysis

In this section, we show that the RIG algorithm is asymptotically optimal for stationary modular objective functions, time-varying modular objective functions, and submodular objective functions. The asymptotic optimality requires that a *conservative* pruning strategy is used that does not remove trajectories that could potentially lead to the most informative plan.

²We note that our proposed algorithm can also be utilized if a fixed goal point is specified. In this case, nodes would enter the closed list when the minimum cost to reach the goal exceed the remaining budget.

We first state a number of modest assumptions that are required for this analysis. These assumptions are adapted from the Rapidly-exploring Random Belief Tree (RRBT) [4] and the Rapidly-exploring Random Graph (RRG) [12] algorithms.

Assumption 1: Let x^a , x^b , and x^c be three points within radius Δ of each other. Let the path e_1 be generated by $Steer(x^a, x^c, \Delta)$, e_2 be generated by $Steer(x^a, x^b, \Delta)$, and e_3 be generated by $Steer(x^b, x^c, \Delta)$. If $x^b \in e_1$, then the concatenated path $[e_2, e_3]$ must be equal to e_1 and have equal cost and information.

This assumption is nearly equivalent to the assumption required for RRBT [4] except that we require that both the cost and information be equal for the concatenated edge. The assumption states that the $Steer()$ function is consistent for intermediate points and that both the cost and information functions are also consistent. This assumption is necessary due to the refinement as additional samples are added, which in the limit leads to samples that are infinitely close together. For general robot dynamics, this property requires that we can simulate the continuous system for any intermediate time step.

Assumption 2: There exists a constant $\delta \in \mathbb{R}_+$ such that for any point $x^a \in \mathcal{X}_{free}$ there exists an $x^b \in \mathcal{X}_{free}$, such that (i) the ball of radius δ centered at x^a lies inside \mathcal{X}_{free} and (ii) x^a lies inside the ball of radius δ centered at x^b .

This assumption is equivalent to the corresponding assumption in the RRG and RRT* algorithms [12] that requires that some free space is available around the optimal trajectory to allow for convergence.

We now examine requirements for pruning strategies that lead to asymptotic optimality of the RIG algorithm.

Theorem 1: Given two nodes n_a and n_b that are co-located with different cost and information values. Let p_a^g be the maximally informative partial trajectory originating from the node corresponding to n_a that satisfies the remaining budget. Similarly, let p_b^g be the maximally informative partial trajectory originating from the node corresponding to n_b that satisfies the remaining budget. If a pruning strategy is employed that removes co-located nodes that are dominated by a partial ordering $n_a < n_b$, RIG is asymptotically optimal if the following condition holds:

$$n_a < n_b \Rightarrow I(p_a^g) + I(n_a) < I(p_b^g) + I(n_b). \quad (2)$$

Proof: (Sketch) The condition above states that a partial ordering exists such that the most informative final trajectory from n_a is always less than the most informative final trajectory from n_b . Thus, no maximally informative trajectory will ever be pruned as long as the partial ordering is upheld. We must now show that the resulting graph will contain the optimal path in the limit. This claim follows from Assumptions 1 and 2 and the analysis of the asymptotic optimality of RRG in [12]. The key idea is that if there is sufficient space to sample within a ball around all points in \mathcal{X}_{free} then, in the limit, an infinitely dense graph will be created within the ball around all points. If the $Near()$ function returns every vertex within a ball of radius $r \propto (\log(n)/n)^{1/d}$, where n is the

number of vertices in the graph and d is the dimensionality of the state, then using Assumption 2, we have that such a ball exists around all points. Additionally, Assumption 1 states that the `Steer()` actions can be infinitely subdivided and hence in the limit will produce all paths that may be optimal. ■

D. Pruning of Suboptimal Plans

In order to ensure that we generate the maximally informative trajectory in the limit, we need to guarantee that we do not remove any partial plans that may lead to the optimal plan. When a new node is added to the graph, all near nodes are extended towards it. For a dense graph, the new node will be reached by multiple nodes in the graph, which will lead to multiple new partial plans with co-located endpoints. When this occurs, it is advantageous to prune partial plans that cannot lead to the optimal solution. We now describe conservative pruning strategies to this effect.

For a modular objective function, it is straightforward to show that if n_a and n_b are co-located, and $c(n_a) > c(n_b)$, where $c(n_a)$ is the cost to reach n_a and $c(n_b)$ is the cost to reach n_b , then $I(p_a^g) \leq I(p_b^g)$. Thus, we can prune paths at a given node if an alternative path exists to that node such that $I(n_a) > I(n_b)$ and $c(n_a) \leq c(n_b)$. Figure 2 shows that pruning suboptimal paths reduces the number of nodes required by more than an order of magnitude for the modular case.

For submodular objective functions, the pruning strategy above may remove partial plans that lead to the optimal plan. To see this, note that the maximally informative trajectory from a node $I(p_a^g)$ depends on the prior path taken to reach that node. Thus, we cannot guarantee that paths should be pruned even if they are dominated both in information and in cost. To achieve a conservative pruning strategy in this case, we must generate an upper bound on $I(p_a^g)$. We can do this by calculating the reachable information $I^{max}(n_a)$, i.e., the information within the remaining budget [2]. The reachable information can be calculated either through integration, sampling, or by enumeration if the information function is evaluated at discrete points. We note that if the reachable information is calculated as an approximation, the pruning strategy must be adjusted accordingly to preserve asymptotic optimality.

Given a way to calculate the reachable information, we can now prune nodes that are co-located if $I(n_a) > I^{max}(n_b)$ and $c(n_a) \leq c(n_b)$. In practice, the more aggressive pruning strategy for modular objectives can alternatively be used as a heuristic for submodular objectives to sacrifice asymptotic optimality for computational efficiency.

V. EXPERIMENTAL RESULTS

We now discuss simulations and field tests validating the proposed informative motion planning algorithm. We compare the RIG algorithm to a combinatorial branch and bound algorithm from prior work [2]. The branch and bound algorithm incrementally extends partial paths in a discrete space and maintains an upper bound on the solution quality of each partial path using the bounding strategy for calculating

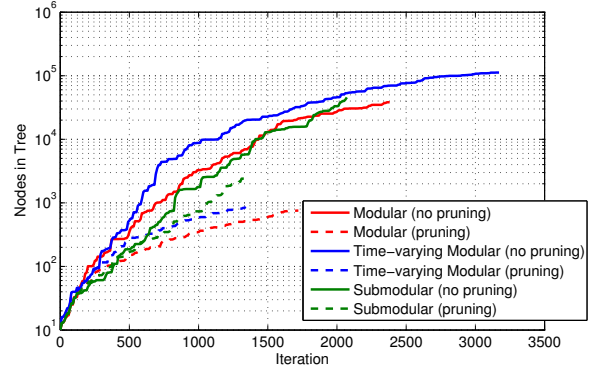


Fig. 2. Number of iterations versus number of nodes in the graph while running the RIG algorithm. The plots are terminated after the optimal solution is found. Pruning paths that are known to be suboptimal can lead to more than an order of magnitude reduction in the number of nodes.

$I^{max}(\cdot)$ described above. The branch and bound algorithm is guaranteed to terminate with the optimal solution. As such, it provides an apt comparison to validate the scalability improvements of RIG as well as the advantage of operating in continuous space.

A. Synthetic Problems

The simulations were run on a single desktop with a 3.2 GHz Intel i7 processor with 9 GB of RAM. The RIG algorithm and combinatorial branch and bound algorithm were both implemented in C++ on Ubuntu Linux. Nearest neighbor queries were provided by the Open Motion Planning Library (OMPL) [7].

The simulations model a synthetic information gathering problem in a $10 \text{ km} \times 10 \text{ km}$ environment. To provide a comparison with the branch and bound algorithm, the vehicle is given a simple motion model where it is assumed to move holonomically on a discrete 1 km grid. The RIG algorithm was modified such that nodes could only be created on the vertices of the 1 km grid, which effectively yields a discrete variant of the algorithm. In some simulations, circular obstacles are placed randomly in the environment with radii (also chosen randomly) between 1 km to 5 km.

The information objective for these simulations was specified by the random placement of five gaussian information sources. The intensity of each information source degrades exponentially with a randomly chosen length scale and intensity. In the time-varying case, the information sources move with a known trajectory. For the submodular case, no additional information is gathered from a point once it has been observed by the vehicle. An example problem and solution trajectory is shown in Figure 1, and an accompanying supplemental video shows animations of growth over time for this example.

Given the specifications described above, 100 random scenarios were chosen for each budget, and the RIG algorithm was compared to the combinatorial branch and bound algorithm. Figure 3 shows the time to find the optimal discrete path for increasing budget lengths. For the modular and time-varying modular objectives, the RIG algorithm is able to find

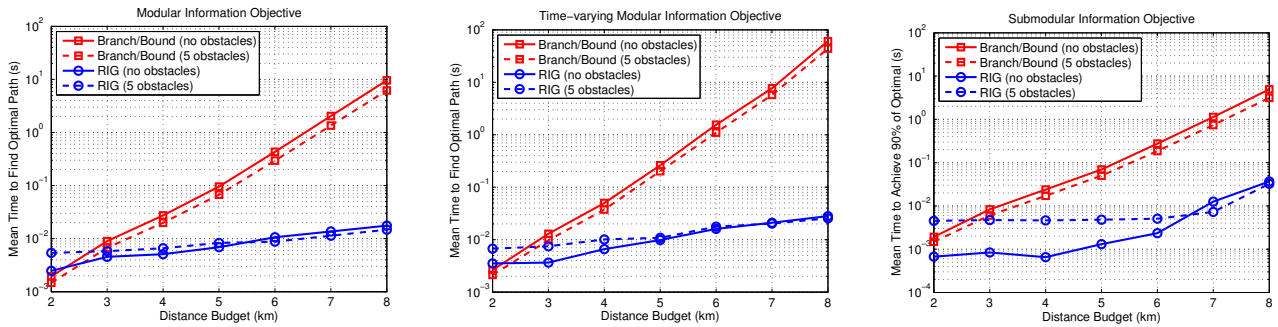


Fig. 3. Simulated results for the proposed sampling-based information gathering algorithm in a 10 km \times 10 km environment. Each data point is averaged over 100 simulated trials with a random information landscape derived using a mixture of Gaussians. For comparison to the discrete branch and bound algorithm, waypoints are restricted to fall on a 1 km grid. For modular and time-varying modular information objectives, the proposed RIG algorithm quickly approaches the optimal discrete solution even at high budgets. For submodular objective functions, RIG quickly approaches a near-optimal solution.

the optimal solution quickly even at higher budgets. Here, the optimal solution was known from running the branch and bound algorithm. In general, the optimal solution can be identified at the point where the graph stops improving. Refining an upper bound on optimal as the algorithm progresses is a potential area for future work.

Adding obstacles to the environment can either increase or decrease the computation time. Obstacles restrict the number of possible paths through the environment, but they also can prevent the algorithm from converging quickly. With larger budgets, the computational gains from restricting the number of paths outweigh the additional convergence time as obstacles are added, and with smaller budgets the opposite occurs.

The gain in computation time for RIG over the combinatorial algorithm is less pronounced for the submodular information objective, due to the looser pruning criterion necessary to guarantee convergence to the optimal solution. Initial simulations (not shown) demonstrated that RIG provides a marginal improvement over the combinatorial algorithm for finding the optimal solution. However, a strength of RIG is that it rapidly explores the environment and quickly generates a near-optimal solution. Figure 3 shows significant improvement in computation time for achieving a solution within 90% of optimal. In the next section, we will also demonstrate how using the RIG algorithm in continuous space can be used to improve path planning for optimizing a submodular information objective in a wireless signal strength mapping application.

B. Wireless Signal Strength Mapping

We now demonstrate our proposed approach using a lake monitoring scenario with an autonomous surface vehicle (ASV). The ASV is propeller-driven and is capable of moving at speeds up to 2 knots. It is equipped with a GPS unit and a Doppler Velocity Log (DVL), which provide localization capabilities for the vehicle. The vehicle communicates with a ground station through a standard 802.11 wireless connection. The vehicle is capable of measuring the wireless signal strength at its current location in dBm. Experiments were conducted at Puddingstone Lake in San Dimas, CA

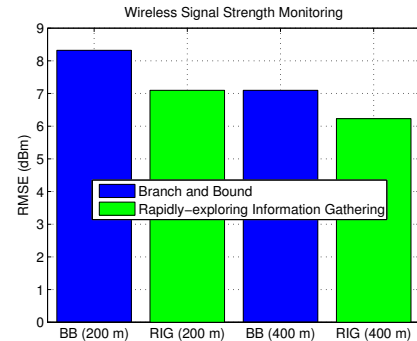


Fig. 4. Root means square error (RMSE) of the predicted wireless signal strength in a lake after executing four different data collection trajectories. The trajectories planned by the branch and bound algorithms are based on a discrete grid, and the trajectories planned by the proposed RIG algorithm operate in continuous space (trajectories are shown in Figure 5). The RIG algorithm is able to achieve lower RMSE for a given trajectory length.

(Lat. 34.088854 $^{\circ}$, Lon. -117.810667 $^{\circ}$).

The vehicle is maneuverable enough to accurately track waypoints given by a planner. As a result, we are able to generate trajectories in 2D space, which significantly reduces the complexity of the planning problem versus planning in the full 6D space of the vehicle’s position, orientation, and velocities. Turning restrictions were incorporated into the `Steer()` function to ensure that the trajectories are executable.³

The goal in this experiment is to reconstruct the wireless signal strength over the entire area of interest, which would be useful for constructing an ad hoc network and planning surfacing locations for underwater exploration missions. In the context of informative motion planning, we want to minimize the root mean square error (RMSE) of the wireless signal strength predictions over the area of interest given a budget on the trajectory length. We note that there is no straightforward way to calculate the expected RMSE after executing a given trajectory due to the difficulty in predicting fluctuations in

³We note that it may be possible to generate a trajectory in 6D that gathers more information for a given trajectory length than the optimal trajectory in 2D. Further analysis of full 6D planning with RIG is an interesting avenue for future work.

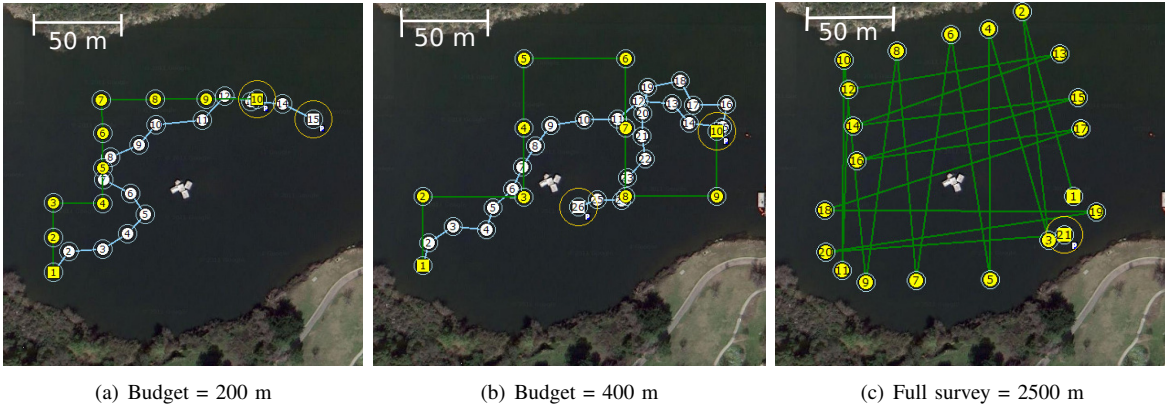


Fig. 5. **Left and Center:** Trajectories taken by the branch and bound algorithm (shown as yellow waypoints) versus the RIG algorithm (shown as white waypoints) by an autonomous vehicle monitoring wireless signal strength in a lake. The RIG algorithm is able to operate in continuous space to provide a richer set of possible trajectories for monitoring the environment. **Right:** The full survey is also shown (planned by a human operator), which was used to compare the predictive accuracy after executing the autonomously planned trajectories.

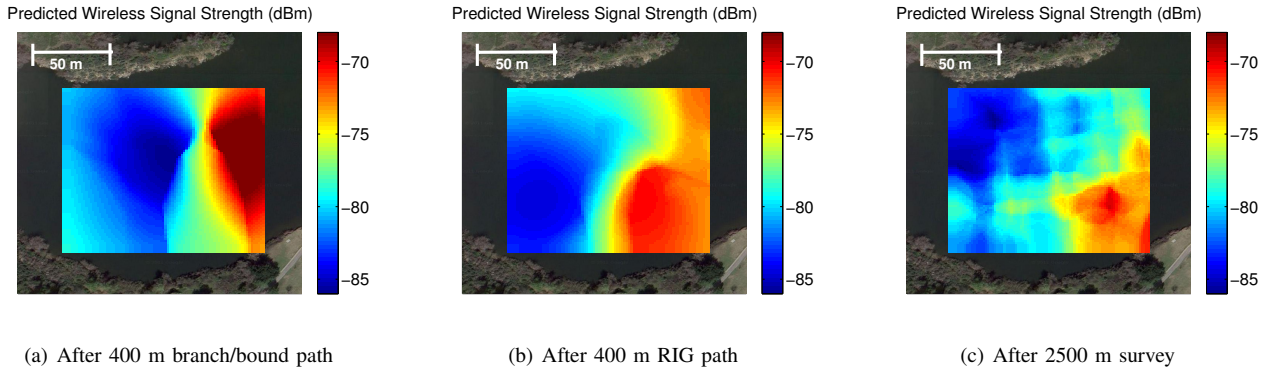


Fig. 6. **Left and Center:** Wireless signal strength map built after executing the 400 m trajectories generated by branch and bound and RIG. **Right:** Wireless signal strength map built after executing the full 2500 m survey. The map generated by RIG is qualitatively closer to the one generated by the full survey.

signal strength. As an alternative, we use a surrogate metric for planning that correlates with RMSE.

We utilize nonparametric regression in the form of Gaussian Processes (GPs) to provide a prediction of the wireless signal strength across the area of interest. For simplicity, we employ a standard squared exponential kernel [17] that captures the fact that predictions that are nearer to each other are more correlated than those that are further apart. We use the predicted variance from the GP as a proxy for RMSE, which has been shown in prior work to correlate [22]. Due to the scalability issues of the GP, a local approximation is employed where only the 100 nearest data points are used to calculate the prediction and variance. For a given trajectory, we can now calculate the expected reduction in variance over some sampled set of points in the space of interest. Using this expected reduction in variance as an information metric fully defines an informative motion planning problem.

Trajectories were planned for budgets of 200 m and 400 m using RIG and the discrete branch and bound algorithm (shown in Figure 5). RIG was run for one minute using the aggressive pruning strategy that prunes all paths that are dominated in both distance and information. We note that the

variance reduction information objective is not modular, which means this pruning strategy is a heuristic here. The branch and bound algorithm was run on the finest grid possible that would lead to completion in less than one minute. The trajectories were then uploaded to the vehicle and executed.

The mean reconstruction errors after executing the different trajectories are shown in Figure 4. For both the 200 m and 400 m budgets, the trajectory generated by the proposed RIG algorithm provides lower RMSE error than the trajectory generated by the discrete branch and bound algorithm. The resulting wireless signal strength maps (shown in Figure 6) also show a qualitative improvement for the paths generated by RIG. These results demonstrate the benefit of planning in continuous space. The experiments on the autonomous surface vehicle also demonstrate the ease of implementation of the proposed algorithm on a fielded system.

VI. CONCLUSIONS AND FUTURE WORK

We have shown that it is possible to generate highly informative motion plans while respecting a budget constraint through the use of iterative sampling-based motion planning algorithms. Variants of the proposed RIG algorithm apply to

modular, time-varying modular, and submodular information objectives. We have also shown that the RIG algorithm is asymptotically optimal, in that it approaches the optimal solution with increasing runtime, as long as a conservative pruning strategy is used to eliminate suboptimal plans. We have demonstrated through simulations that the proposed algorithm is capable of finding optimal solutions in discrete domains quickly. We have also demonstrated the effectiveness of the RIG algorithm in a wireless signal strength mapping domain where it is able to improve the accuracy of a wireless signal strength map given a limited budget on mission time.

The general algorithm proposed in this work opens up a number of avenues for future work. More aggressive pruning strategies may allow for asymptotically optimal behavior with submodular objective functions while pruning away more suboptimal trajectories. Additional work could also provide more efficient methods for calculating the information of a given trajectory for complex objective functions. More future work lies in generating more intelligent sampling strategies that focus the candidate trajectories towards particularly informative areas of the environment. Finally, the proposed algorithm can potentially be extended to incorporate objective functions that do not have the property of submodularity. The work we have presented here provides a foundation for the application of sampling-based motion planning algorithms to a wide range of robotic information gathering tasks.

ACKNOWLEDGMENTS

The authors gratefully acknowledge Jonathan Binney from Willow Garage for assistance with data processing. This research has been funded in part by the following grants: ONR N00014-09-1-0700, ONR N00014-07-1-00738, ONR N00014-06-10043, NSF CCR-0120778 and NSF CNS-1035866.

REFERENCES

- [1] R. Bajcsy. Active perception. *Proc. IEEE, Special Issue on Computer Vision*, 76(8):966–1005, Aug. 1988.
- [2] J. Binney and G. S. Sukhatme. Branch and Bound for Informative Path Planning. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 2147–2154, May 2012.
- [3] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte. Coordinated decentralized search for a lost target in a Bayesian world. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 48–53, Oct. 2003.
- [4] A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 723–730, May 2011.
- [5] A. Cameron and H. Durrant-Whyte. A Bayesian approach to optimal sensor placement. *Int. J. Robotics Research*, 9(5):70–88, Oct. 1990.
- [6] S. Chen, Y. Li, and N. M. Kwok. Active vision in robotic systems: A survey of recent developments. *Int. J. Robotics Research*, 30(11):1343–1377, Sept. 2011.
- [7] I. A. Şucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Mag.*, 19(4):72–82, Dec. 2012.
- [8] G. Hollinger, S. Singh, J. Djugash, and A. Kehagias. Efficient multi-robot search for a moving target. *Int. J. Robotics Research*, 28(2):201–219, Feb. 2009.
- [9] G. Hollinger, B. Englot, F. Hover, U. Mitra, and G. Sukhatme. Active planning for underwater inspection and the benefit of adaptivity. *Int. J. Robotics Research*, 32(1):3–18, Jan. 2013.
- [10] K. Hsiao, L. P. Kaelbling, and T. Lozano-Pérez. Task-driven tactile exploration. In *Proc. Robotics: Science and Systems Conf.*, June 2010.
- [11] S. Patil J. van den Berg and R. Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *Int. J. Robotics Research*, 31(11):1263–1278, Sept. 2012.
- [12] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *Int. J. Robotics Research*, 30(7):846–894, June 2011.
- [13] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller. Anytime motion planning using the RRT*. In *IEEE Int. Conf. Robotics and Automation*, pages 1478–1483, May 2011.
- [14] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robotics and Automation*, 12(4):566–580, Aug. 1996.
- [15] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *Int. J. Robotics Research*, 20(5):378–400, 2001.
- [16] D. S. Levine. Information-rich path planning under general constraints using rapidly-exploring random trees. Master’s thesis, Massachusetts Institute of Technology, June 2010.
- [17] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.
- [18] J. H. Reif. Complexity of the mover’s problem and generalizations. In *Proc. IEEE Symp. Foundations of Computer Science*, pages 421–427, Oct. 1979.
- [19] A. Ryan and J. K. Hedrick. Particle filter based information-theoretic active sensing. *Robotics and Autonomous Systems*, 58(5):574–584, May 2010.
- [20] A. Singh, A. Krause, C. Guestrin, and W. Kaiser. Efficient informative sensing using multiple robots. *J. Artificial Intelligence Research*, 34:707–755, Apr. 2009.
- [21] R. Smith, M. Schwager, S. Smith, B. Jones, D. Rus, and G. Sukhatme. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *J. Field Robotics*, 28(5):714–741, Sept. 2011.
- [22] S. Vasudevan, F. T. Ramos, E. W. Nettleton, and H. F. Durrant-Whyte. Gaussian process modeling of large scale terrain. *J. Field Robotics*, 26(10):812–840, Oct. 2009.
- [23] A. Wald. Sequential tests of statistical hypotheses. *Ann. Mathematical Statistics*, 16(2):117–186, June 1945.