Efficient Graph-Based Informative Path Planning Using Cross Entropy

Junghun Suh, Kyunghoon Cho, and Songhwai Oh

***Abstract*— In this paper, we present a novel informative path planning algorithm using an active sensor for efficient environmental monitoring. While the state-of-the-art algorithms find the optimal path in a continuous space using sampling-based planning method, such as rapidly-exploring random graphs (RRG), there are still some key limitations, such as computational complexity and scalability. We propose an efficient information gathering algorithm using an RRG and a stochastic optimization method, cross entropy (CE), to estimate the reachable information gain at each node of the graph. The proposed algorithm maintains the asymptotic optimality of the RRG planner and finds the most informative path satisfying the cost constraint. We demonstrate that the proposed algorithm finds a (near) optimal solution efficiently compared to the state-of-the-art algorithm and show the scalability of the proposed method. In addition, the proposed method is applied to multi-robot informative path planning.**

I. Introduction

In recent years, environmental monitoring has become increasingly important due to factors, such as global warming, ozone layer depletion, deforestation, ocean pollution, natural resource depletion, and population growth, to name a few. A number of different environmental monitoring problems have been studied extensively, including marine monitoring [1], ecological monitoring [2], aerial monitoring [3], and disaster monitoring [4], [5].

In order to monitor a large area, it is important to collect the most useful information with available sensing resources. Guestrin et al. [6] proposed a method for placing static sensors using the concept of entropy by modeling the environmental parameters using Gaussian processes. However, it requires a dense deployment of sensors to avoid sensing holes and static sensors are not suitable for time-varying processes we often find in nature. To overcome the limitations of static sensors, mobile sensor networks are introduced to increase the sensing coverage both in space and time.

With mobile sensor networks, active sensing is possible by taking advantage of the mobility of sensing agents. We will call the problem of scheduling the trajectory of a mobile sensor for collecting the most useful information as an informative path planning (IPP) problem in this paper.

Singh et al. [4] presented an efficient informative planning strategy for maximizing the mutual information from a team of robots. In [7], an optimal IPP algorithm using branch and bound was proposed to maximally reduce the variance of the estimate about the field of interest based on exhaustive search. However, aforementioned methods were applied to discretized search spaces, making it less scalable for large problems. A sampling based path planning method, a rapidly-exploring random tree (RRT) [8], has been applied to an IPP problem. Kwak et al. [9] applied a genetic algorithm to decide which node of an RRT tree to extend for information gathering. A mobile target tracking controller was developed to maximize the information gain using a limited number of mobile sensors in [10], where the tracking accuracy along the path represents the information gain. While the recently proposed RRT* [11] seems like a good candidate for solving an IPP problem, Bry et al. [12] has shown that RRT* is not suitable for solving an IPP problem with a cost constraint. While RRT* performs rewiring with the assumption that the optimal cost to the goal region from a node is independent from the cost used to reach the node, the optimal information gain to the goal from a node depends on the path taken to the node and the cost constraint.

Motivated by this fact, Hollinger et al. [13] proposed a rapidly-exploring information gathering (RIG) algorithm, which combines sampling-based motion planning with combinatorial optimization. Since it follows the overall structure of rapidly-exploring random graphs (RRG) [11], it can ensure that the optimal path will be found as the number of samples approaches infinity. It introduced a pruning strategy to improve the efficiency by reducing co-located nodes if they cannot lead to the optimal solution. The pruning strategy requires to know the upper bound on the possible reachable information gain using the given cost budget for each node. The reachable information gain is used as the upper bound and it is computed using a branch and bound algorithm [7]. However, while robots move in a continuous space, the reachable information gain is calculated by discretizing the state space. Hence, it requires heavy computation to perform the branch and bound algorithm to compute all reachable information gains for all nodes in a tree for a complex or high dimensional problem.

In this paper, we propose an RRG-based planner, called CE-IPP, which extends the RIG planner [13]. CE-IPP uses cross entropy (CE) [14], a stochastic optimization method, to efficiently estimate the reachable information gain. The CE framework allows us to choose a (near) optimal informative trajectory among trajectory samples satisfying the cost budget constraint. The proposed algorithm guarantees the asymptotic optimality like RRGs and ensures to find a (near) optimal informative path to the goal region which satisfies the cost budget constraint, if such a path exists. For each node in the graph, the proposed algorithm updates the lower bound of the reachable information gain of the most informative path to the goal region, which passes through the node. As more nodes are added to the graph, the lower bound of the reachable information gain of each node improves and converges to the maximum value in the limit.

This research was supported in part by the Brain Korea 21 Plus Project in 2016 and by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2009348).

J. Suh, K. Cho, and S. Oh are with the Department of Electrical and Computer Engineering and ASRI, Seoul National University, Seoul 151-744, Korea (e-mail: {junghun.suh, kyunghoon.cho, songhwai.oh}@cpslab.snu.ac.kr).

The remainder of this paper is structured as follows. In Section II, we describe the optimal informative path planning problem with a cost budget. CE-IPP is presented in Section III-A. Numerical simulation results are discussed in Section IV.

II. Problem Formulation

Let $\mathcal{Q}$ be a region, in which a robot (or a mobile sensor) performs its assigned sensing tasks. Let $\mathcal{X} \subset \mathbb{R}^n$ be the state space of the mobile sensor, where $\mathcal{X} = \mathcal{X}_{free} \cup \mathcal{X}_{obs}$. We denote $\mathcal{X}_{obs}$ and $\mathcal{X}_{free}$ as the obstacle region and the obstacle free space, respectively, where $\mathcal{X}_{free} = \mathcal{X} \setminus \mathcal{X}_{obs}$. A state $x \in \mathcal{X}$ includes the position $q \in \mathcal{Q}$. We assume in this paper that the motion of the mobile sensor is deterministic. Let Σ be a set of all collision-free paths, such that, for $\mathcal{P} \in \Sigma$, $\mathcal{P}(t) \in \mathcal{X}_{free}$ for $t \in [0, T]$, where $\mathcal{P}$ is the trajectory of a mobile sensor and T is the termination time. Let $\Omega \subset \Sigma$ be a set of all feasible paths, such that, for $\mathcal{P} \in \Omega$, $\mathcal{P}(0) = x_{init}$ and $\mathcal{P}(T) \in x_{goal}$, where $x_{init} \in \mathcal{X}$ is the initial state and $x_{goal} \subset \mathcal{X}$ is the goal region. Let $c : \Sigma \to \mathbb{R}^+$ be a cost function, e.g., the path length or the energy consumption of a robot following the path. We assume that the cost function returns strictly positive value for any collision-free path and it is monotonic, additive, and bounded.

An IPP problem finds a path with the maximum information gain obtained through entropy [15], mutual information [6] or variance reduction [16] criterion while considering the cost of the path. We can formulate an IPP problem as the following optimization problem by placing the cost of a path as a budget constraint:

$$\arg\max_{\mathcal{P} \in \Omega} \ \mathcal{I}\Big(\mathcal{P}\Big) \quad \text{subject to } c(\mathcal{P}) \ \leq \ \mathcal{B}, \tag{1}$$

where the function $\mathcal{I} : \Sigma \to \mathbb{R}^+$ returns the information collected along the path and $\mathcal{B} \in \mathbb{R}^+ < \infty$ is a budget for the maximum allowed cost. Generally, $\mathcal{I}$ is submodular defined on a finite set, i.e., it has the diminishing return property. It is known that the problem of finding the optimal solution of a discrete version of (1) is NP-hard [17], [18]. Hence, a greedy algorithm is used in practice by discretizing the search space. If a path is defined in a continuous space, then solving (1) becomes more challenging. Note that we assume that a continuous path is discretized into a set of finite number of points, where the robot senses, to evaluate $\mathcal{I}$ along a continuous path.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ are finite sets of vertices and edges, respectively. We denote $\mathcal{P}_a^b$ a trajectory initiated at a and arrived at b (i.e., $\mathcal{P}_a^b(0) = a$ and $\mathcal{P}_a^b(T) = b$). Consider two partial path $\mathcal{P}_{a_1}^{b_1}$ and $\mathcal{P}_{a_2}^{b_2}$. If $\mathcal{P}_{a_1}^{b_1}(T) = \mathcal{P}_{a_2}^{b_2}(0)$, then we denote $\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}$ as the concatenated trajectory, i.e.,

$$\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}(t) := \begin{cases} \mathcal{P}_{a_1}^{b_1}(\frac{t}{\alpha}) \text{ for all } t \in [0, \alpha T], \\ \mathcal{P}_{a_2}^{b_2}(\frac{t-\alpha T}{1-\alpha}) \text{ for all } t \in (\alpha T, T], \end{cases}$$

where $0 < \alpha < 1$. Given a vertex $v \in \mathcal{V}$, a path via v can be defined as follows.

Definition 1: A v-trajectory $\mathcal{P}^v$ is a feasible concatenated trajectory $\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}$, such that $\mathcal{P}_{a_1}^{b_1}(T) = \mathcal{P}_{a_2}^{b_2}(0) = v$, $\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}(0) = x_{init}$, $\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}(T) \in x_{goal}$, and $c(\mathcal{P}_{a_1}^{b_1}|\mathcal{P}_{a_2}^{b_2}) = c(\mathcal{P}_{a_1}^{b_1}) + c(\mathcal{P}_{a_2}^{b_2}) \leq \mathcal{B}$.

With a slight abuse of notation, we will allow $v \in \mathcal{V}$ to be used in place of v's corresponding state $x \in \mathcal{X}$. Let $\{\mathcal{P}_{ALG}^v\}$ be a set of all v-trajectories via the vertex v, which can be produced by an algorithm ALG. ALG is a label indicating any sampling based algorithm with a graph structure. Then we denote $\mathcal{P}_{ALG,n}^v$ as the n-th v-trajectory among $\{\mathcal{P}_{ALG}^v\}$. For any fixed vertex $v \in \mathcal{V}$, we use $g_{v,n} = \mathcal{I}(\mathcal{P}_{ALG,n}^v)$ to denote the amount of information obtained along $\mathcal{P}_{ALG,n}^v$. Then the maximum of $g_{v,n}$, g_v^*, can be formulated as follows:

$$g_v^* = \limsup_{n \to \infty} \ g_{v,n}. \tag{2}$$

Since the algorithm constructs a graph $\mathcal{G}$, a path from x_{init} to v for $v \in \mathcal{V}$ can be obtained from the graph, so it estimates $\mathcal{P}_v^{x_{goal}}$, which is a partial path of $\mathcal{P}_{ALG}^v$. If there exists a vertex v which gives no $\mathcal{P}_v^{x_{goal}}$ due to the budget cost, g_v at v is set to 0. We can guarantee that $g_v^* < \infty$ for all $v \in \mathcal{V}$. Since the cost function returns monotonically positive value for any collision-free path, any v-trajectory has a finite length, so the amount of information is bounded. Let $\mathcal{V}_i^{ALG}$ be a set of vertices in the graph obtained by ALG at the end of iteration i. Assuming that we can update $g_{v,n}$ at any vertex $v \in \mathcal{V}_i^{ALG}$, then we can have g_v^* for all $v \in \mathcal{V}_i^{ALG}$ as $i \to \infty$, since $n \to \infty$ as $i \to \infty$, i.e.,

$$\limsup_{i \to \infty} \left(\max_n \ g_{v,n} \right) = g_v^*. \tag{3}$$

The optimal informative path planning problem asks for finding a feasible path with maximum g_v^*, so we can approach the problem as finding

$$g^*(i) = \max_{v \in \mathcal{V}_i^{ALG}} \left(\max_{1 \leq m \leq n_v(i)} g_{v,m} \right), \tag{4}$$

where $n_v(i)$ is the number of v-trajectories after iteration i, as we iterate to add more vertices to the graph. From the vertices $v^*(i)$ with information gain $g^*(i)$, we can find a maximal informative trajectory by connecting $v^*(i)$. In this paper, we propose an incremental graph structure based algorithm that utilizes a stochastic optimization method to generate informative trajectories satisfying constraints in terms of the cost budget. This stochastic optimization based method allows for the rapid generation of near optimal trajectories even for complex information quality objectives.

III. IPP Using Cross Entropy

This section deals with the application of the CE method to informative path planning problem. We present the CE based informative path planning (CE-IPP) algorithm which finds a path to the goal region while maximizing information gain within the budget constraint. CE-IPP is based on an RRG for optimal planning to the goal region and the cross entropy method for solving maximization problem with the inequality constraint.

A. CE-IPP Algorithm

Algorithm 1 describes the main body of the proposed CE-IPP algorithm. The algorithm shares the overall structure with RRGs and the following procedures of RRGs, which

are $Sample(\cdot)$, $Nearest\_Neighbor(\cdot)$, $Steer(\cdot)$, $Near(\cdot)$, and $CollisionFree(\cdot)$. However, since the cost constraint is considered in RRGs, we slightly modify an RRG by adding the extra procedure as the graph grows. This procedure follows the approach presented in [13]. Whenever a new node is added, edges are created between the new node and nearby nodes. Since there exists the budget constraint, we need to check the cost integrated along the trajectory from x_{init} to the end node of a newly added edge (i.e., a nearby node or new node) in order to determine whether to insert the edge to the graph. If the cost is over the budget, then the corresponding edge is deleted. However, unlike [13], where each vertex in the graph has only possible reachable information quantity, each vertex in a CE-IPP graph has a path to the goal and the maximum information gain along the path. The near-optimal path at each vertex is found by cross entropy path planning considering the cost-to-go value, i.e., the remained budget at the node, if there exists a feasible path. On account of the additional procedure, each vertex stores the most informative path via the vertex and the information gain collected along the path. Since CE-IPP follows RRGs and finds the near-optimal path at each vertex while satisfying the budget constraint, it provides not only a guarantee of asymptotic optimality but also an efficient approach for finding a solution even if any node in the graph has not reached the goal region.

The graph $\mathcal{G}$ initially contains only one vertex v_{init} as an initial node, where the information gain $v_{init}.g$ and the path $v_{init}.\mathcal{P}^{x_{init}}$ of v_{init} are initialized. The algorithm terminates once stopping criterion satisfies the condition. The stopping criterion can be the number of iterations after reaching the goal region from any node in graph, the maximum number of iterations, or a time deadline. At each iteration, the graph grows by drawing a random sample x_{rand} and then trying to extend toward x_{rand} from its nearest vertices $v_{nearest}$ as in the RRG algorithm (lines 4–6). A set of v_{near} in the graph that are close to x_{new} are returned as V_{near} through the $Near$ function (line 7). If the line segment connecting between a near node and the new node is not in collision with obstacles, then an edge is inserted into the graph and $Update\_Bound$ function is called (lines 10–21). $Update\_Bound$ updates the most informative path to the goal via node v and the maximum value of the information gain by finding a v-trajectory through the $CE\_Estimate$ function.

Two critical functions of CE-IPP, $Update\_Bound$ and $CE\_Estimate$, are described below.

1) Update_Bound: The $Update\_Bound$ procedure is described in Algorithm 2 and graphically illustrated in Figure 1. Whenever a new feasible vertex v_{new} is selected, co-located nodes can be generated at v_{new} and v_{near}. Figure 1 shows multiple co-located nodes $v_{new}.x_i$ of the vertex v_{new} (e.g., $v_{new}.x_1, v_{new}.x_2, v_{new}.x_3$). These co-located nodes consist of vertices which have the same location but different paths from x_{init} to x_{new}. Thus, each remaining budget, $\mathcal{B}_{v_{new}.x_i}$, at co-located nodes is determined according to cost of each $\mathcal{P}^{v_{new}.x_i}_{v_{init}.x}$. Moreover, the same procedure is done for co-located nodes of v_{near}. For all co-located nodes at each vertex, $Update\_Bound$ function is performed. After computing the remained budget $\mathcal{B}_{x'_i}$ using $\mathcal{C}_{x'_i}$, where $\mathcal{C}_{x'_i}$ is the cost along the path $\mathcal{P}^{x'_i}_{v_{init}.x}$, $CE\_Estimate$ finds a near optimal trajectory $\mathcal{P}^{x_{goal}}_{x'_i}$ initiated at x'_i to the goal region within $\mathcal{B}_{x'_i}$ and computes the information gain $\mathcal{I}_{x'_i}$ along $\mathcal{P}^{x'_i}_{x_{init}} \cup \mathcal{P}^{x_{goal}}_{x'_i}$ (lines 3–5). If a path cannot be found at x'_i due to the budget constraint, $CE\_Estimate$ function returns an empty $\mathcal{P}^{x_{goal}}_{x'_i}$ and a zero gain for $\mathcal{I}_{x'_i}$. Then we drop the node and repeat $CE\_Estimate$ for a different co-located node x'_i.

The lower bound $v'.g$ on a possible information gain along paths via v' is updated by computing information gains for all co-located v' and it is used to solve (4). The number of co-located nodes increases much faster than the increase in the number of vertices in the graph. Hence, there can be issues

Algorithm 1 CE-IPP

Input: 1) Start position x_{init} and goal position x_{goal}
2) Budget $\mathcal{B}$
3) A ball of radius r for neighbors

Output: The most informative path from x_{init} to x_{goal} within $\mathcal{B}$

1: $v_{init}.x \leftarrow x_{init}, v_{init}.g \leftarrow 0, v_{init}.\mathcal{P}^{x_{init}} \leftarrow \emptyset$
2: $\mathcal{V} \leftarrow \{v_{init}\}, \mathcal{E} \leftarrow \emptyset, \mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$
3: **while** stopping_criterion is false **do**
4: $x_{rand} \leftarrow$ a random sample from $\mathcal{Q}$
5: $v_{nearest} \leftarrow$ Nearest_Neighbor$(\mathcal{V}, x_{rand})$;
6: $x_{new} \leftarrow$ Steer$(v_{nearest}.x, x_{rand})$
7: $V_{near} \leftarrow$ Near(x_{new}, r)
8: $v_{new}.x \leftarrow x_{new}, v_{new}.g \leftarrow 0, v_{new}.\mathcal{P}^{v_{new}} \leftarrow \emptyset$
9: $\mathcal{V} \leftarrow \mathcal{V} \cup \{v_{new}\}$
10: **for** all $v_{near} \in V_{near}$ **do**
11: **if** CollisionFree$(v_{near}.x, v_{new}.x)$ **then**
12: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(v_{near}.x, v_{new}.x)\}$
13: $v_{new} \leftarrow$ Update_Bound(v_{near}, v_{new})
14: **end if**
15: **end for**
16: **for** all $v_{near} \in V_{near}$ **do**
17: **if** CollisionFree$(v_{new}.x, v_{near}.x)$ **then**
18: $\mathcal{E} \leftarrow \mathcal{E} \cup \{(v_{new}.x, v_{near}.x)\}$
19: $v_{near} \leftarrow$ Update_Bound(v_{new}, v_{near})
20: **end if**
21: **end for**
22: $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$
23: **end while**
24: Find v which satisfies (3)

Algorithm 2 Update_Bound(v, v')

Output: v'

1: **for** all $x_i \in v.x$ **do**
2: $\mathcal{C}_{x'_i} \leftarrow \mathcal{C}_x + c(x_i, v'.x)$
3: $\mathcal{B}_{x'_i} \leftarrow \mathcal{B} - \mathcal{C}_{x'_i}$
4: $\langle \mathcal{P}^{x_{goal}}_{x'_i}, \mathcal{I}_{x'_i} \rangle \leftarrow$ CE_Estimate$(x', x_{goal}, \mathcal{P}^{x'_i}_{v_{init}.x}, \mathcal{B}_{x'_i})$
5: **if** $\mathcal{I}_{x'_i} > g_{x'}$ **then**
6: $v'.g \leftarrow \mathcal{I}_{x'_i}, v'.\mathcal{P}^{x'} \leftarrow \mathcal{P}^{x'_i}_{x_{init}} \cup \mathcal{P}^{x_{goal}}_{x'_i}$
7: **end if**
8: **end for**

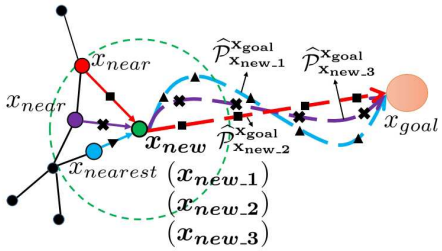


Fig. 1. Update procedure at x_{new}. This figure shows the different paths initiated at multiple co-located nodes for x_{new} in the graph. Once x_{new} is inserted into the graph, since all edges from three neighbor vertices are added, there exists multiple co-located nodes of the vertex x_{new}. For each vertex in graph, $Update\_Bound$ function determines the best information quantity along each trajectory estimated at co-located nodes (e.g., $\widehat{\mathcal{P}}^{x_{goal}}_{x_{new\_i}}$ for $i = 1, 2, 3$ at x_{new}).

with time and space complexities. One of solution is to apply the pruning procedure of [13], so we can ignore unnecessary co-located nodes with no possibility of increasing the lower bound of the information gain.

2) CE_Estimate: The $CE\_Estimate$ function is a special case of CE path planning [19]. The function returns a probabilistically near-optimal trajectory from a vertex v to the goal region, in terms of the information gain along $\mathcal{P}^v$, using the remained budget $\mathcal{B}_v.x$.

Unlike [19], which focuses on samples minimizing a cost function, $CE\_Estimate$ generates samples which maximize the information gain considering the budget constraint and it can be formulated as follows:

$$\max_{\mathcal{P}^{x_{goal}}_v \in \Omega} \mathcal{I}(\mathcal{P}^v_{x_{init}} \cup \mathcal{P}^{x_{goal}}_v) \quad \text{subject to } c(\mathcal{P}^{x_{goal}}_v) \leq \mathcal{B}_v, \tag{5}$$

where $\mathcal{P}^v_{x_{init}} \cup \mathcal{P}^{x_{goal}}_v = \mathcal{P}^v$ and $\mathcal{B}_v = \mathcal{B} - c(\mathcal{P}^v_{x_{init}})$. As shown in (5), the $CE\_Estimate$ function needs $\mathcal{P}^v_{x_{init}}$ as an input to find the near-optimal trajectory from v to the goal. Since the information function used in this paper is submodular, the prior trajectory to v is involved in determining $\mathcal{P}^{x_{goal}}_v$. By considering $\mathcal{P}^v_{x_{init}}$ as a part of the trajectory $\mathcal{P}^v$, the submodular property is satisfied. It is important to generate initial trajectory samples from the probability density function $p(\cdot; \theta_k)$ at $k = 0$ which cover the search space well. In this work, it is assumed that $p(\cdot; \theta_k)$ follows the multivariate Gaussian distribution with parameter $\theta_k = (\mu_k, \Sigma_k)$ as done in [19]. By adjusting the covariance Σ_0 to cover the region of interest, we can get those initial samples. However there exists a limitation in initial sampling due to the budget constraint. The coverage and the budget are in conflict with each other (i.e., for a good coverage, more budget is required and the coverage can be poor for a small budget). Thus, we need to deal with good coverage against the budget constraint. In order to perform the initial sampling efficiently, we find a feasible region satisfying the budget for sampling. As explained in [19], the parameter θ_k consists of m primitives. In this section, we assume that the primitives and the budget represent waypoints and the length of the path, respectively. We also assume that the given cost function is defined in the Euclidean space, which represents the length of the path. Given any two points v and x_{goal}, we assume that the path $\mathcal{P}^{x_{goal}}_v$ passes via $x \in \mathcal{X}_{free}$. Then the cost of $\mathcal{P}^{x_{goal}}_v$ via x, $f(x)$, is equal to the cost of $\mathcal{P}^x_v$, $g(x)$, plus the cost of $\mathcal{P}^{x_{goal}}_x$, $h(x)$, and $f(x)$ should satisfy the remained budget constraint at v, $\mathcal{B}_v$, i.e.,

$$f(x) = g(x) + h(x) \leq \mathcal{B}_v. \tag{6}$$

The possible set of states that satisfies (6) can then be expressed as

$$\mathcal{X}_v = \{x \in \mathcal{X}_{free} | \|v - x\|_2 + \|x_{goal} - x\|_2 \leq \mathcal{B}_v\} \tag{7}$$

which is a general equation for an n-dimensional prolate hyperspheroid, a special hyperellipsoid. The focal points are v and x_{goal}, the transverse diameter is $\mathcal{B}_v$, and the conjugate diameter is $\sqrt{\mathcal{B}_v^2 - L^2}$, where $L = \|v - x_{goal}\|_2$ [20].

In order to draw x from $p(\cdot; \theta_k)$ within the feasible region, we use the truncated multivariate normal distributions based on $\mathcal{X}_v$ defined as follows:

$$f(x|\mu, \Sigma, \mathbf{a}, \mathbf{b}) \propto \exp\Big\{ -\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu) \Big\} \tag{8}$$

for $\mathbf{a} \leq x \leq \mathbf{b}$, and, otherwise, 0, where $\mathbf{a} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{R}^n$ are the lower and upper limits of $\mathcal{X}_v$, respectively. One approach for generating variates from a truncated multivariate normal distribution is to use the Gibbs sampler [21]. Since the Gibbs sampler generates an instance from the univariate distribution conditional on the current values of other variables, the sample x is obtained through the Gibbs sampler by generating each component successively, i.e., $x_i \sim P(x_i | x_{\neg i})$, for $i = 1, \ldots, n$, where $x_{\neg i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ [21]. For all m waypoints as primitives, we sample each waypoint sequentially to make a path via all m waypoints. Thus we can generate a set of N trajectory samples $\{\mathcal{P}^{x_{goal}}_{v,i}\}_{i=1}^N$ which satisfies the budget constraint $\mathcal{B}_v$. Once trajectory samples satisfying the budget constraint are generated, the parameter update procedure is performed to find the optimal θ^* as explained in [14]. However, trajectory samples with more information quantity are selected as the elite samples to update parameters in this work. By iterating the above procedures until convergence, we can solve (5).

In order to show that CE-IPP can produce the optimal solution as the number of samples goes to infinity, we should obtain g^*_v at all $v \in \mathcal{V}^{ALG}_i$ as $i \to \infty$. It can be shown by the following lemma.

Lemma 1: Let $\Omega_{v,\mathcal{B},i}$ denote the set of v-trajectories, where v is contained in the graph built by CE-IPP at iteration i for budget $\mathcal{B}$ and $\Omega_{v,\mathcal{B}}$ be a set of all v-trajectories satisfying the budget $\mathcal{B}$. Then, $(\lim_{i\to\infty} \Omega_{v,\mathcal{B},i}) = \Omega_{v,\mathcal{B}}$

Proof: The proof of this lemma follows directly from Lemma 4.4 in [13]. It is shown that all feasible trajectories for budget $\mathcal{B}$ can be generated if the algorithm builds an infinitely dense connected graph and the length of all feasible trajectories obtained from the algorithm is finite. Since CE-IPP follows the RRG scheme, it constructs an infinitely dense connected graph [13]. In addition, any path included in $\Omega_{v,\mathcal{B},i}$ has a finite length as described in Section II. Hence, CE-IPP generates all feasible trajectories for budget $\mathcal{B}$. Therefore, for each v, we have the desired result. ■

IV. Experiments

We show the effectiveness of the proposed algorithm using numerical simulations of single robot and multiple robot informative path planning problems. All simulations were implemented in MATLAB and run on a computer with a 3.4GHz Intel Xeon processor and 8 GB RAM.

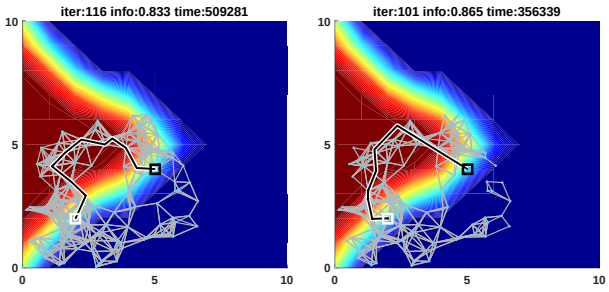
A. Single Robot Informative Path Planning

We compared the proposed algorithm against RIG-graph [13]. Since both algorithms show the asymptotic optimality, we examine how efficiently each algorithm finds a better solution. First, we generated two variance maps which represent simple and complex initial uncertainties over a 10 km $\times$ 10 km sensing field by deploying different numbers of static sensors.[1] The Gaussian process MATLAB toolbox [22] is used to model the uncertainty map. We assume that we can obtain measurements from static sensors at all times. For fair comparison, the simple map is made similarly to the map used in [13] and it is shown in Figure 2. Regions with static sensors have low uncertainties and are represented in blue color and the red region represents high uncertainty. We use a point-mass vehicle and assume that it takes measurements at every 1 km intervals as it moves in a continuous space. In order to capture the informativeness of a trajectory, we choose to maximize the reduction in variance of the field as the information function, as similarly done in [7]. Since the selected information function is submodular, it requires a discretized grid space as an input, so we convert the path found in the continuous space to corresponding points in a 11×11 grid by choosing the closest grid point. Furthermore, we assume that taking multiple measurements at the same location does not affect the collected information quantity, as done in [13].

Since there exists a fixed goal region, we limit the graph extension when the remaining budget of a node exceeds the shortest distance from the node to the goal region. Figure 2 shows a visual comparison of RIG-graph and CE-IPP, showing maximum informative trajectories found by two algorithms. Since the proposed algorithm tries to estimate a near-optimal path at each node of the graph, more computation is required at each iteration. However, CE-IPP returns a good initial informative path more efficiently than RIG-graph, since it can estimate the path using the remained budget at each node even if the graph has not reached the goal region. Moreover, the initially obtained path from CE-IPP is more informative than the results obtained from RIG-graph with a much longer running time. The results from five simulations using different pre-specified random seeds are shown in Figure 3(a), showing the average maximum information gain found by each algorithm as a function of the running time of each algorithm. The shaded region indicates twice the standard deviation at each time. In all trials, the proposed algorithm outperforms RIG-graph.

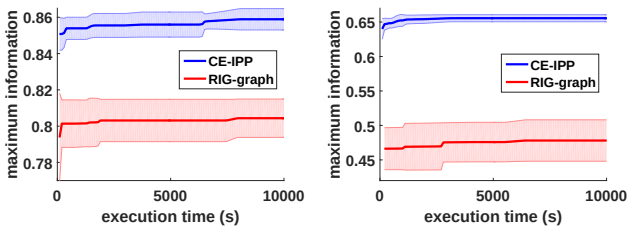
We increased the budget to 20 km to cover the whole region in case of the complex map. For this longer budget, the performance is more distinguishable as shown in Figure 3(b) since trajectories obtained from CE-IPP passes by covering high uncertainty regions over the whole search space. As mentioned earlier, RIG-graph computes the maximally reachable information gain based on a reachable region within the allowable budget, when performing the pruning procedure. However, the reachable region is the same for many co-located nodes when a large budget is given, thereby causing pruning procedure ineffective.

[1]For the simple map, 74 static sensors are placed with a uniform separation. For the complex map, 17 sensors are placed randomly.



(a) RIG-Graph (b) CE-IPP

Fig. 2. Graphs and maximum informative trajectories obtained from (a) RIG-graph and (b) CE-IPP. Each vertex is represented by a cyan dot. White and black rectangles represent a start point and goal region, respectively. A black solid line represents the maximum informative path at the current iteration. The budget constraint for this scenario is 8 km.



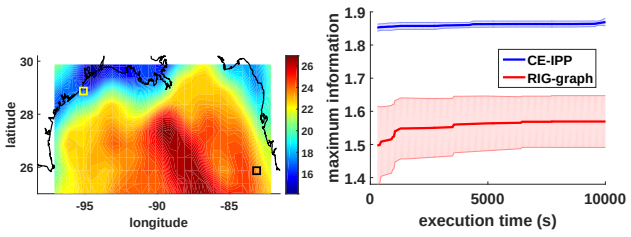
(a) Simple map (b) Complex map

Fig. 3. The average maximum information gain of a path as a function of running time for a number of simulations. The shaded area indicates twice the standard deviation of those runs. (a) Simple uncertainty map. (b) Complex uncertainty map.

We also tested two algorithms on the sea surface temperature field of Gulf of Mexico as shown in Figure 4(a). High temperature regions are represented in red and low temperature regions are shown in blue. The grid size of the map is scaled to 31×11 and no initial sensors are placed. The average results from 10 independent runs are shown in Figure 4(b). Again, the proposed algorithm finds near-optimal solutions more efficiently than RIG-graph.

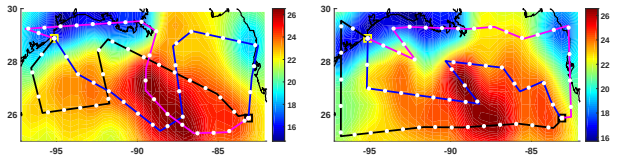
B. Multi-Robot Informative Path Planning

An optimal approach to perform multi-robot planning is to consider all robots simultaneously in a product space of all robots. However, in practice, it is not a feasible solution due to its exponentially increased complexity. In order to demonstrate the efficiency of the proposed method, we perform multi-robot informative path planning in a product space and compare to a greedy planner, in which each robot is planned sequentially using the proposed algorithm to improve the overall information gain. The proposed method on the product space and the sequential greedy planner are



(a) (b)

Fig. 4. IPP on the sea surface temperature field of Gulf of Mexico. (a) Temperature field. (b) The average maximum information gain of a path as a function of running time.



(a) Greedy planner (b) Product space planner

Fig. 5. Trajectory results obtained by the greedy planner and the product space planner are shown in (a) and (b), respectively, using three agents

tested on the sea surface temperature field of Gulf of Mexico. We let both versions of algorithms run for the same time by making sure that the overall run time of sequential planning of the greedy planner matches the run time of the product space planner.

We compared results with different numbers of robots. Figure 5(a) and Figure 5(b) show trajectories obtained by two different planners for three agents, respectively. The temperature field is estimated based on measurements along the path. Table I shows the root mean square error of the temperature field estimations for each case. From Table I, we know that the estimated fields from the product space planner are more accurate than the greedy planner. As shown in Figure 5(a), more overlapping trajectories are found by the greedy planner, wasting novel sensing opportunities compared to the product space planner. This example demonstrates the efficiency of the proposed method and its applicability for efficient multi-robot informative path planning.

V. Conclusions

In this paper, we have presented a new approach to solve an informative path planning problem over a continuous space for environmental monitoring. The proposed algorithm combines a sampling based planning method and stochastic optimization method. It guarantees the asymptotic optimality, but, in addition, it can also return a near-optimal informative path at any node of the constructed graph if there exists a path satisfying the cost budget constraint. We have shown that the proposed algorithm finds a near optimal solution more efficiently and scales well compared to the state-of-the-art algorithm for informative path planning. The proposed method is also applied to multi-robot informative path planning and has shown performance benefits compared to a greedy planner.

# of robots	The greedy planner	The product space planner
2	2.4200	2.1419
3	0.8886	0.5984

TABLE I

The root mean square error (RMSE) of the temperature field estimations.

References

[1] N. E. Leonard, D. A. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. E. Davis, "Collective motion, sensor networks, and ocean sampling," *Proceedings on the IEEE*, vol. 95, 2007.

[2] N. Cao, K. H. Low, and J. M. Dolan, "Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms," in *Proc. of the IEEE International Conference on Autonomous agents and multi-agent systems*, 2013.

[3] R. Marchant and F. Ramos, "Bayesian optimisation for informative continuous path planning," in *Proc. of the IEEE International Conference on Robotics and Automation*, Hong Kong, China, May 2014.

[4] A. Singh, A. Krause, and W. J. Kaiser, "Nonmyopic adaptive informative path planning for multiple robots," in *Proc. of the International Joint Conference on Artificial Intelligence*, 2009.

[5] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A multi-robot control policy for information gathering in the presence of unknown hazards," in *Proc. of the International Symposium on Robotics Research*, 2011.

[6] C. Guestrin, A. Krause, and A. P. Singh, "Near-optimal sensor placements in gaussian processes," in *Proc. of the International Conference on Machine learning*, 2005.

[7] J. Binney and G. S. Sukhatme, "Branch and bound for informative path planning," in *Proc. of the IEEE International Conference on Robotics and Automation*, Saint Paul, USA, May 2012.

[8] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, no. 3, pp. 378–400, May 2001.

[9] J. Kwak and P. Scerri, "Path planning for autonomous information collecting vehicles," in *Proc. of the International Conference on Information Fusion*, 2008.

[10] D. Levine, B. Luders, and J. P. How, "Information-rich path planning with general constraints using rapidly-exploring random trees," in *Proc. of AIAA Infotech@Aerospace Conference*, Atlanta, GA, April 2010.

[11] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[12] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *Proc. of the IEEE International Conference on Robotics and Automation*, San Diego, CA, June 2011.

[13] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *The international Jounal of Robotics Research*, vol. 32, no. 12, pp. 1271–1287, 2014.

[14] R. Y. Rubinstein, "Optimization of computer simulation models with rare events," *European Journal of Operational Research*, vol. 99, no. 1, pp. 89–112, 1997.

[15] C. Ko, J. Lee, and M. Queyranne, "An exact algorithm for maximum entropy sampling," *Operations Research*, vol. 43, no. 4, pp. 684–691, 1995.

[16] A. Krause, H. B. McMahan, C. Guestrin, and A. Gupta, "Robust submodular observation selection," *Journal of Machine Learning Research*, vol. 9, pp. 2761–2801, Dec 2008.

[17] A. Krause, B. McMahan, A. Gupta, and J. Kleinberg, "Near-optimal sensor placements:maximizing information while minimizing communication cost," in *Proc. of the International Symposium on Information Processing in Sensor Networks*, 2006.

[18] A. Krause and C. Guestrin, "Nonmyopic active learning of gaussian processes:an exploration-exploitation approach," in *Proc. of the International Conference of Machine Learning*, 2007.

[19] M. Kobilarov, "Cross-entropy randomized motion planning," in *Proc. of the Robotics: Science and Systems*, Los Angeles, USA, June 2011.

[20] J. D. Gammelland, S. S. Srinivasaand, and T. D. Barfoot, "Informed RRT*: Optimal incremental path planning focused through an admissible ellipsoidal heuristic," *arXiv preprint arXiv:1404.2334*, 2014.

[21] C. P. Robert, "Simulation of truncated normal variables," *Statistics and computing*, vol. 5, no. 2, pp. 121–125, 1995.

[22] C. E. Rasmussen and C. K. Williams, Eds., *Gaussian Processes for Machine Learning*. Cambridge: The MIT Press, 2006.