

No-Regret Replanning under Uncertainty

Wen Sun¹, Niteesh Sood², Debadeepta Dey³, Gireeja Ranade³,
Siddharth Prakash², and Ashish Kapoor³

Abstract—This paper explores the problem of path planning under uncertainty. Specifically, we consider online receding horizon based planners that need to operate in a latent environment where the latent information can be modeled via Gaussian Processes. Online path planning in latent environments is challenging since the robot needs to explore the environment to get a more accurate model of latent information for better planning later and also achieves the task as quick as possible. We propose UCB style algorithms that are popular in the bandit settings and show how those analyses can be adapted to the online robotic path planning problems. The proposed algorithm trades-off exploration and exploitation in near-optimal manner and has appealing no-regret properties. We demonstrate the efficacy of the framework on the application of aircraft flight path planning when the winds are partially observed.

I. INTRODUCTION

Finding an optimal path under unknown or partially observed environment is a challenging and an important task in robotics. In this paper, we consider an online replanning framework where in each round, the robot picks a direction to traverse and as it travels, it receives observations about unknown variables along the trajectory. The robot then considers this newly acquired information to refine its knowledge about the environment, which in turn influences the action selection in the next round. Finding an optimal strategy is challenging in such online replanning framework as the robot essentially faces a tradeoff between exploration and exploitation. In order to make inferences about the latent variables, the robot needs to pick actions that can drive itself around the space to gather information. Such exploration can be beneficial as more accurate knowledge of the environment promises more accurate estimation of the cost of trajectories. However, such actions come at a cost, especially if these information foraging actions make the robot deviate from its mission. Thus, it is important for the robot to make the right decisions about when/where to explore and when to exploit.

We specifically focus on receding-horizon replanning, where the robot is equipped with a pre-computed library of trajectories and planning entails picking a trajectory among the library at every round. Also, we consider the cases where the uncertainties in environment are unknown but can be approximately modeled by Gaussian Processes. There is a fairly large and important classes of natural phenomenon, including winds, oceanic currents and traffic volume that is spatially correlated, can be modeled with GPs. For example,

consider an aerial robot that is attempting to minimize traversal time by exploiting the tail winds while avoiding the head winds. However, without complete information it is difficult for the robot to make the optimal decision. Consequently, we ask how should the aircraft traverse in the space in order to maximally utilize the winds while continuously sensing and updating its belief about the wind field.

This paper addresses such exploration-exploitation trade-off in the online replanning framework by presenting a new and simple replanning strategy called *Upper Confidence Bound* Replanning (UCB-Replanning). When trading between exploration and exploitation, UCB-Replanning uses the classic strategy of *Optimism in the Face of Uncertainty*. Since Gaussian Processes provide a distribution over the latent variable of interest, UCB-Replanning can leverage the inferred uncertainty to build confidence intervals on the estimations of the cost of trajectories. During replanning, the robot then takes the estimation of the cost and the confidence interval of the estimation together into consideration to make a decision. We further analyze the performance of UCB-Replanning. Particularly, we show that UCB-Replanning is *no-regret* in a sense that the UCB-Replanning in average is doing almost as well as picking the optimal trajectory assuming the full knowledge of unknown variables, along the states that the robot has visited.

Finally we conduct a case study of aircraft navigating under wind uncertainty, where the wind speed is modeled by a Gaussian Process. We investigate the experimental performance of UCB-replanning under different types of wind map, including wind maps over the continental United States, constructed from real wind data provided by National Oceanic and Atmospheric Administration (NOAA).

II. RELATED WORK

We discuss UCB-Replanning in relation to literature in three main areas: 1. Receding-horizon planning in robotics 2. Partially Observable Markov Decision Processes (POMDPs) and 3. Multi-Armed Bandit problems (MAB).

Receding-horizon planning: In receding-horizon control a library of pre-computed control command sequences are simulated forward from the current state of the robot using the dynamic motion model to come up with a set of dynamically feasible trajectories up to the planning horizon. This set of trajectories is then evaluated on the map of the world in the vicinity of the robot and amongst all the currently collision-free trajectories the one that makes most progress towards the goal is chosen for traversal [1]. The selected trajectory is traversed for a portion of the time and the process of trajectory

¹The Robotics Institute, School of Computer Science, Carnegie Mellon University. The research work was done during an internship at Microsoft Research, Redmond. wensun@cs.cmu.edu.

²Microsoft Research, India

³Microsoft Research, Redmond

evaluation and selection is repeated again. Receding-horizon based planning has been widely used in aerial and ground robot navigation in cluttered environments [2], [3], [4] due to many attractive properties like finite runtime, adaptability to available computational budget and dynamic feasibility by construction. We use receding-horizon planning with pre-computed trajectory libraries as the framework in this paper.

Partially Observable Markov Decision Processes (POMDPs): POMDPs are used to model Markov Decision Processes (MDPs) where only part of the state of the world can be observed. Finding optimal policies of POMDP is NP-hard [5]. Approximate solutions like Point-Based Value Iteration [6], [7], Heuristic Search-Based Value Iteration [8] and Monte-Carlo planning [9] are popular goal-free reward oriented solvers. While goal-oriented methods like [10] are more relevant to our problem scenario, they are hard to adapt to continuous observation spaces and computation and time budgets imposed by mobile robots. Belief Space Planning approaches (e.g., [11], [12], [13], [14], [15], [16]) is also related to our work. But most of belief space planning approaches assume that the uncertainty is known, e.g., the form of the stochastic dynamics are fully known. We do not even assume the form of the uncertainty is known here and we utilize Gaussian Process to keep tracking the uncertainty in an online manner while the robot is moving.

Our work is closely related to that of Dey et al., [17] who combined Canadian Traveler Problem (CTP) with GPs to formulate the problem of replanning as a Gaussian Traveler Problem (GTP). GTPs used determinization techniques like hindsight optimization [18] to efficiently incorporate uncertainty over all edge costs of the graph in a GTP and show lower empirical cost of traversal to goal for an aircraft navigating partially known wind fields over continental US than merely replanning by the mean prediction over edge costs. GTPs have a number of limitations: 1. Discretization effects due to representing the problem on a graph. Making the graph dense has negative computational effects. 2. The edges of the graph may not be dynamically feasible for the aircraft to track. 3. The hindsight optimization determinization step requires sampling large number of possible future graph states which can be expensive. UCB-Replanning mitigates all of these issues.

Multi-Armed Bandits: *Optimism in the face of uncertainty* is a classic strategy for trading off between exploration and exploitation in many Multi-Armed Bandit problems [19], [20], [21], [22] and Reinforcement Learning (RL) problems [23], [24]. The classic Upper Confidence Bound (UCB) algorithm for MAB [19], [22] maintains a confidence interval of the true reward for each arm and pull the arm with the maximum upper bound of its confidence interval. UCB-Replanning leverages the classic analysis of MAB to analyze the performance of online receding horizon based planning.

III. PRELIMINARIES

Let us define $\mathbb{X} \subset \mathbb{R}^d$ as the state space for the robot. The state $\mathbf{x} \in \mathbb{X}$ includes the information of the robot such as positions and velocities. We model the uncertainty in the

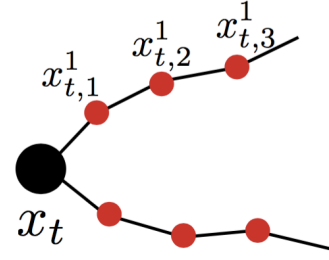


Fig. 1: Notation of waypoints (red circles) of a set of trajectory library. At time step t the robot is located at the root (black circle) of the trajectories and needs to make a decision about which trajectory to traverse next.

environment by a random variable $v \in \mathbb{R}$.¹ The realization of random variable v depends on state of the robot and is modeled by an *unknown* function subject to noise:

$$v = g(\mathbf{x}) + \epsilon, \quad (1)$$

where we assume $\epsilon \sim \mathcal{N}(0, \sigma^2)$. These random variable could encode the variant types of uncertainties in the environment such as the speed of the wind at the current position of the robot, the estimated distance to a obstacle and so on. For notation simplicity, in the rest of the work, we define $v(\mathbf{x})$ as a (noisy) realization of the random variable v at state \mathbf{x} . Throughout this work, we assume that the unknown g is sampled from a Gaussian process prior $\text{GP}(0, \kappa(\mathbf{x}, \mathbf{x}'))$. Given a set of pairs $\{\mathbf{x}_i, v(\mathbf{x}_i)\}_{i=1}^N$, the posterior over g is GP distribution with mean $\mu(\mathbf{x})$, covariance $\text{cov}(\mathbf{x}, \mathbf{x}')$ and variance $\sigma^2(\mathbf{x})$ as:

$$\begin{aligned} \text{cov}(\mathbf{x}, \mathbf{x}') &= \kappa(\mathbf{x}, \mathbf{x}') - \kappa_N(\mathbf{x})^T (\mathbf{K}_N + \sigma^2 \mathbf{I})^{-1} \kappa_N(\mathbf{x}'), \\ \mu(\mathbf{x}) &= \kappa_N(\mathbf{x})^T (\mathbf{K}_N + \sigma^2 \mathbf{I})^{-1} \mathbf{y}_N, \quad \sigma^2(\mathbf{x}) = \text{cov}(\mathbf{x}, \mathbf{x}), \end{aligned}$$

where $\kappa_N(\mathbf{x}) = [\kappa(\mathbf{x}_1, \mathbf{x}), \dots, \kappa(\mathbf{x}_N, \mathbf{x})]^T$ and \mathbf{K}_N is the gram matrix with $\mathbf{K}_N[i, j] = \kappa(\mathbf{x}_i, \mathbf{x}_j)$.

We assume that the robot is equipped with a pre-computed library of trajectories $\{\tau_1, \dots, \tau_K\}$, where each trajectory τ_i consists of L segments (Fig. 1). Assume that at time step t , the robot's state is denoted as \mathbf{x}_t . Starting at \mathbf{x}_t , for each trajectory τ_i , the $L + 1$ waypoints are represented as $\{\mathbf{x}_{t,0}^i, \mathbf{x}_{t,1}^i, \dots, \mathbf{x}_{t,L}^i\}$, where $\mathbf{x}_{t,0}^i = \mathbf{x}_t$, for all $i \in \{1, 2, \dots, K\}$ (Fig. 1).

At step t , located at state \mathbf{x}_t , the robot needs to pick a trajectory indexed at $I_t \in \{1, 2, \dots, K\}$ and execute τ_{I_t} . Then the robot will traverse along τ_{I_t} and end at state $\mathbf{x}_{t,L}^{I_t}$. At the beginning of next around $t + 1$, we set $\mathbf{x}_{t+1} = \mathbf{x}_{t,L}^{I_t}$, and repeat the above process. At every step t , each trajectory τ_i is equipped with a reward function $f_{t,i}$, which measures the reward of executing trajectory τ_i at step t . The reward function depends on the uncertain variables v along the trajectory and we denote $f_{t,I_t}(\{v(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) : \mathbb{R}^{L+1} \rightarrow \mathbb{R}$. Throughout this paper, we assume that $f_{t,i}$ is Lipschitz constant with respect to ℓ_1 norm with Lipschitz constant l .

¹It is straightforward to extend to multi-variable case where variables can be modeled by multiple independent GPs

The ideal goal of the robot is to pick a sequence of trajectories $\{I_1^*, \dots, I_T^*\}$ from iteration $t = 1$ to T , so that it can maximize the cumulative reward $\sum_{t=1}^T f_{t,I_t}(\{g(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L)$ (Here we focus on maximizing with respect to $g(\mathbf{x})$, which is the expectation of $v(\mathbf{x})$). Note that computing the optimal sequence of decisions $\{I_1^*, \dots, I_T^*\}$ requires the full knowledge of the underlying map g , which is not available.

It is not easy for the robot to pre-plan a sequence of the decisions since the robot does not have the exact information about g except a prior, which could be non-informative. To refine its knowledge about $g(\mathbf{x})$, the robot needs to explore the area near \mathbf{x} to collect observations of v and update the GP. Hence the robot needs to plan on the fly while collecting new information and refining its knowledge about g for future planning. The robot essentially faces the tradeoff between exploration and exploitation: the robot needs to explore by choosing difference trajectories to get the information about the uncertain variables at different regions of its state space while it also needs to exploit by picking temporally high-reward trajectories to maximize its total reward.

IV. ALGORITHM

We leverage the strategy of optimism in the face of uncertainty to design our algorithm for robot to perform online replanning. Especially, we design an algorithm that is similar to UCB, where we maintain a confidence interval of the true reward for each trajectory. To design the confidence interval for each trajectory's reward at every step, we first extract a confidence interval of the uncertain variable v from GP. We then use the Lipschitz continuity of the reward function of each trajectory to transfer the confidence interval of the uncertain variable v to the confidence interval of the reward of each trajectory. We finally choose the trajectory with the highest upper confidence bound of the reward estimation. The detailed algorithm is presented in Alg. 1. In round t , Alg. 1 first use the current GP model $(\mu_{t-1}, \sigma_{t-1})$ to compute the means and the standard deviations of v along the waypoints of all K trajectories (Line. 4 and 5). Then for each trajectory τ_k , Alg. 1 using the Lipschitz constant (l) and a scaling parameter β_t (will be defined later in analysis) to compute the upper confidence bound of the reward function as shown in Line 6. It then picks the trajectory τ_{I_t} that has the highest upper confidence bound. During the execution of τ_{I_t} , the robot receives observations of v along the waypoints and online updates the GP model (Line. 9).

A. Analysis

We analyze the performance of Alg. 1. Particularly, we are interested in analyzing the *regret*, which measures the difference between Alg. 1's cumulative reward and the cumulative reward if one always picks the best trajectories along the states that the robot traversed when executing Alg. 1. More formally, let us assume that the the sequence of states that the robot visited at all T rounds as: $\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ and the indexes of the trajectories that the robot picked at all T

Algorithm 1 UCB-Replanning

- 1: **Input:** A library of K trajectories $\{\tau_k\}_{k=1}^K$, sequence of parameters $\beta_t \in \mathbb{R}^+$. A GP (μ_0, σ_0) that models the variable v over the state space \mathbb{X} .
 - 2: **for** $t = 1$ to T **do**
 - 3: **for** $k = 1$ to K **do**
 - 4: Compute the sequence of means of g on the waypoints on τ_k as $\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L$.
 - 5: Compute the sequence of standard deviations of g on the waypoints as $\{\sigma_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L$.
 - 6: Compute the upper confidence bound of the reward function $f_{t,k}$ as: $b_k := f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L) + l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^k)$.
 - 7: **end for**
 - 8: Choose index $I_t = \arg \max_{k \in \{1, \dots, K\}} b_k$ and execute trajectory τ_{I_t} .
 - 9: Observe samples of g along the waypoints as $\{v(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^{L-1}$ and use these L samples to online update GP to obtain μ_t and σ_t .
 - 10: **end for**
-

rounds as $\{I_1, \dots, I_T\}$. We define the regret as:

$$\mathbf{R}_T = \frac{1}{T} \left[\sum_{t=1}^T \max_{I_t^* \in [K]} f_{t,I_t^*}(\{g(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L) - \sum_{t=1}^T f_{t,I_t}(\{g(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) \right] \quad (2)$$

Namely, at each round t , we measure how much more reward the robot could gain if it could pick I_t^* instead of I_t at \mathbf{x}_t . The goal is to make regret converges to zero so that in average the robot has little regret in terms of choosing I_t .

We remark that our regret definition measures the difference between the rewards of an optimal decision maker with full access to latent information and the rewards of the learning algorithm *along the states taken by the learning algorithm*. Ideally one would be interested in the regret of the learning algorithm in respect to the rewards of the optimal decision maker *along the states generated from the optimal decision maker itself*. It turns out that the latter definition of regret is impossible to achieve without any assumptions about the reachability of the systems and the ability to reset [24]. Consider the MDP shown in Fig. 2 with 3 states and 2 actions. Once the agent makes the mistake of taking action a_2 at x_0 , possibly due to the lack of the full knowledge of the model, it will be stuck in x_2 forever and the regret with respect to the optimal decision maker on the optimal sequence $\{x_0, x_1, x_1, \dots\}$ will grow linearly. It is also worth mentioning that our definition of regret in Eqn. 2 is similar to the classic sample complexity definition of exploration in reinforcement learning [24], in the sense that the sample complexity of exploration measures the number of mistakes the learning algorithm makes on the sequence of states generated from the algorithm, instead of the sequence of the states resulting from the optimal policy.

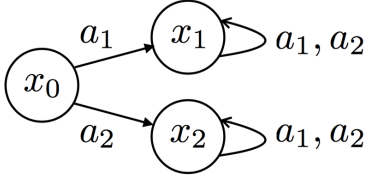


Fig. 2: A decision making problem with three states and two possible actions. The agent starts from x_0 and upon taking action a_1 (a_2) reaches x_1 (x_2). The reward structure is such that the agent receives a reward 1 upon landing at state x_1 and no reward elsewhere. Once the agent lands in either x_1 or x_2 , there is no way for it to move to any other state. Hence the optimal sequence of state for this problem is $\{x_0, x_1, x_1, \dots\}$.

Now we are ready to show that Alg. 1 is no-regret: $\mathbf{R}_t \rightarrow 0$, as $T \rightarrow \infty$. Let us define D_t as the states of all waypoints on all K trajectories: $D_t = \{\{\mathbf{x}_{t,j}^k\}_{j=0}^L\}_{k=1}^K$, where $|D_t| = (L+1)K$. The following lemma builds the confidence interval over g on all waypoints in D_t at all rounds t :

Lemma 1: With $\beta_t = 2 \log(\frac{|D_t|\pi_t}{\delta})$ and any π_t that satisfies $\sum_t 1/\pi_t = 1, \pi_t > 0$, with probability at least $1 - \delta$ we have:

$$\forall t, \forall x \in D_t, |g(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leq \beta_t^{1/2} \sigma_{t-1}(x). \quad (3)$$

The above lemma is essentially the same as Lemma 5.1 in [21]. For completeness we include the proof in the appendix.

The next lemma builds a confidence interval over the rewards of all K trajectories, at all rounds.

Lemma 2: Set $\beta_t = 2 \log(\frac{|D_t|\pi_t}{\delta})$ and $\sum_t 1/\pi_t = 1, \pi_t > 0$, we have with probability at least $1 - \delta$:

$$\begin{aligned} & \forall k \in [K], \forall t, |f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L) - f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L)| \\ & \leq l \beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^k). \end{aligned} \quad (4)$$

Proof: Let us define event A as $\forall t, \forall x \in D_t, |g(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x})$ and from the previous lemma, we know that event A happens with probability at least $1 - \delta$. We now condition on event A . Below we show that event A will imply the inequality in the above lemma.

Since we assume that the reward functions $f_{t,k}$ is Lipschitz continuous, we must have for any t and k :

$$\begin{aligned} & |f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L) - f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L)| \\ & \leq l \sum_{j=0}^L |g(\mathbf{x}_{t,j}^k) - \mu_{t-1}(\mathbf{x}_{t,j}^k)|, \end{aligned}$$

where l is the Lipschitz constant. Conditioned on the fact that event A happens, we have that at any round t and for any trajectory τ_k , we have that:

$$\begin{aligned} & |f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L) - f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L)| \\ & \leq l \sum_{j=0}^L |g(\mathbf{x}_{t,j}^k) - \mu_{t-1}(\mathbf{x}_{t,j}^k)| \leq l \sum_{j=0}^L \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_{t,j}^k), \end{aligned}$$

where we use the assumption that f is l -Lipschitz continuous in ℓ_1 norm. To this end, we have already shown that event A implies the inequality in the above lemma. Since the probability that the event A happens is at least $1 - \delta$, we must have with probability at least $1 - \delta, \forall t, k$

$$\begin{aligned} & |f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L) - f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L)| \\ & \leq l \sum_{j=0}^L \beta_t^{1/2} \sigma_{t-1}(\mathbf{x}_{t,j}^k). \end{aligned} \quad (5)$$

Hence we prove the lemma. \blacksquare

Now we present the main theorem. We consider two types of Kernels: (1) linear kernel as $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$, and (2) square exponential kernel as $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-c\|\mathbf{x} - \mathbf{x}'\|^2)$.

Theorem 3: With $\beta_t = 2 \log(\frac{|D_t|\pi_t}{\delta})$ and $\sum_t 1/\pi_t = 1, \pi_t > 0$, we will have with probability at least $1 - \delta$:

$$\mathbf{R}_T/T \leq O(d \log(LT)/T) \rightarrow 0, \quad T \rightarrow \infty, \quad (6)$$

for linear kernel as $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$, and

$$\mathbf{R}_T/T \leq O((\log(LT))^{d+1}/T) \rightarrow 0, \quad T \rightarrow \infty, \quad (7)$$

for squared exponential kernel $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-c\|\mathbf{x} - \mathbf{x}'\|^2)$.

Proof: [Proof Sketch] For the sake of brevity, we provide the complete proof in the appendix and an abbreviated sketch below. Let us define event B as the inequality 4 shown in Lemma 2. From Lemma. 2 we know that the probability of event B happens is at least $1 - \delta$. Below we show that event B implies the two equalities in the above theorem. For the rest of the proof, we assume we condition on that event B happens. Consider round t . Note that I_t is defined as:

$$\begin{aligned} I_t = \arg \max_{k \in [K]} & f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L) \\ & + l \beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^k), \end{aligned} \quad (8)$$

and I_t^* is defined as:

$$I_t^* = \arg \max_{k \in [K]} f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L), \quad (9)$$

namely the best trajectory one would pick at this round t if g is known. Now let us define the single step regret r_t as:

$$r_t = f_{t,I_t^*}(\{g(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L) - f_{t,I_t}(\{g(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L), \quad (10)$$

namely the regret one has by choosing I_t instead of I_t^* at round t . Similar to classic analysis of UCB based algorithms, we can upper bound r_t using Eqn. 8 and 9:

$$r_t \leq 2l \beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t}). \quad (11)$$

Square both sides of the above inequality and use similar techniques from [21], we have for r_t^2 :

$$\begin{aligned} r_t^2 &= 4l^2 \beta_t (\sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t}))^2 \leq 4l^2 \beta_t L \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2 \\ &\leq 4l^2 \beta_T L \sigma^2 C_1 \sum_{j=0}^L \log(1 + \sigma^{-2} \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2) \end{aligned} \quad (12)$$

where $C_1 = \sigma^{-2}/\log(1 + \sigma^{-2}) \geq 1$.

Since the regret $\mathbf{R}_T = \sum_{t=1}^T r_t$, we must have $\mathbf{R}_T^2 \leq T \sum_t r_t^2$. Using Lemma 5.3 and Lemma 5.4 from [21], we can link \mathbf{R}_T to the maximum information gain as follows:

$$\begin{aligned} \mathbf{R}_T^2 &\leq 4l^2 \beta_T L^2 \sigma^2 C_1 T \sum_{t=1}^T \sum_{j=0}^{L-1} \log(1 + \sigma^{-2} \sigma_{t-1} (\mathbf{x}_{t,j}^{I_t})^2) \\ &\leq 4l^2 \beta_T L^2 \sigma^2 C_1 T \gamma_T, \end{aligned}$$

where γ_T is the maximum information gain defined as

$$\begin{aligned} \gamma_T &= \max_{A \subseteq \mathbb{X}, |A|=LT} I(v_A; g) \\ &= \max_{A \subseteq \mathbb{X}, |A|=LT} H(v_A) - H(v_A|g), \end{aligned}$$

where $H(x)$ is the entropy of the random variable x , $H(x|y)$ is the conditional entropy, $v_A = \{g(\mathbf{x}) + \epsilon\}_{\mathbf{x} \in A}$ is the set of observations of $g(\mathbf{x})$ for all states \mathbf{x} in set A . Namely γ_T quantifies the maximum reduction in uncertainty about g from revealing the observations of g on LT states.

Theorem 5 from [21] shows $\gamma_T \leq O(d \log LT)$ if $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$ and $\gamma_T \leq O((\log LT)^{d+1})$ if $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-c\|\mathbf{x} - \mathbf{x}'\|^2)$. Substitute these results to the above inequality, we prove the theorem. ■

The above theorem shows that as the number of rounds approaches infinity, in average, the policy presented at Alg. 1 performs almost as well as the *best* policy which can always choose the best trajectory at every round. Note that the average regret of using squared exponential kernel (e.g., RBF kernel) shrinks more slowly than the average regret of linear kernel. This indicates that in general if the wind map g is complicated (i.e., requiring RBF kernel to model it), Alg. 1 requires more rounds to achieve good performance.

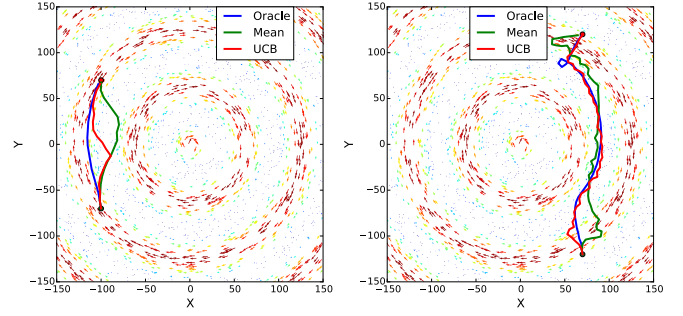
V. CASE STUDY: AIRCRAFT NAVIGATION UNDER WIND UNCERTAINTY

We conduct a case study of aircraft navigation under wind uncertainty and show how UCB-Replanning can be applied. Let us define the state space of the aircraft $\mathbb{X} \in \mathbb{R}^2$ (2D position) where we assume that \mathbb{X} is compact. We fix the norm of airplane's speed to $v_0 \in \mathbb{R}^2$. At a particular position \mathbf{x} , the speed of the wind $v \in \mathbb{R}^2$ is computed from an unknown mapping $g : \mathbb{X} \rightarrow \mathbb{R}^2$, subject to Gaussian noise as $v(\mathbf{x}) = g(\mathbf{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. We assume that range of the speed of wind is bound as $\|v\|_2 \in [v_{\min}, v_{\max}]$, where $v_{\min}, v_{\max} \in \mathbb{R}^+$, and we further assume that $v_{\max} = \|v_0\|_2/2$. Give the position \mathbf{x} of the aircraft, the real speed of the aircraft can be computed as follows:

$$\tilde{v}(\mathbf{x}) = v_0 + \frac{\langle v_0, v(\mathbf{x}) \rangle}{\|v_0\|_2^2} v_0, \quad (13)$$

where the second part of the RHS of the above equation is the projection of the wind speed $v(\mathbf{x})$ at location \mathbf{x} onto the airplane's speed. Overall the goal of the aircraft is to leverage the speed of the wind to decrease its traveling time.

We use Gaussian Process with squared exponential kernel $\kappa(\mathbf{x}, \mathbf{x}') = \alpha \exp(-c\|\mathbf{x} - \mathbf{x}'\|^2)$ ($c, \alpha \in \mathbb{R}^+$) to model the wind speed. At every round, the aircraft chooses a trajectory



(a) Wind field 1 (Tail Wind) (b) Wind field 1 (Head Wind)

Fig. 3: Trajectories (solid lines) resulting from Oracle, Mean and UCB, in the synthetic wind field, with different start (green dot) and goal position (red dot) settings

from a pre-computed library of $K = 48$ trajectories [1]. Each trajectory is a spline with L segments, each segment having a length d .

Given the start position and the final position, the goal of the aircraft is to minimize the total traveling time. Hence our reward function is related to the traveling time. Let us assume that at round t , the robot is at state \mathbf{x}_t . For each trajectory $\tau_k, k \in [K]$, we design the reward function with respect to the speed of the wind as:

$$f_{t,k}(\{v(\mathbf{x}_{t,j}^k)\}_{j=0}^L) = -\left(\sum_{i=0}^{L-1} \frac{d}{\|\tilde{v}(\mathbf{x}_{t,i}^k)\|_2} + \lambda_t t_g\right), \quad (14)$$

where $\lambda_t \in (0, 1]$. The first part of the RHS of the above equation is the total time for traversing the trajectory τ_k while the second part t_g serves as an estimation of the left time for traveling from the end of the trajectory τ_k and the final position. In this work, we use the total time of traveling along the Great Circle Route for t_g (Note that t_g could also be regarded as a scaled shortest distance to goal).²

To apply UCB-Replanning, we first set $\pi_t = \pi^2 t^2 / 6$. It is straightforward to compute the Lipschitz continuous constant l of $f_{t,k}$ as $l = \frac{4d}{\|v_0\|_2^2}$. Hence we can set $l\beta_t^{1/2} = O\left(\frac{4d}{\|v_0\|_2^2} \sqrt{\log(\frac{KL\pi^2 t^2}{\delta})}\right)$. Throughout the experiments, we set $\delta = 0.05$. Namely we want the inequalities in Theorem 3 to hold with probability at least 95%. For specific values of $l\beta_t^{1/2}$, we set $l\beta_t^{1/2} = c \frac{4d}{\|v_0\|_2^2} \sqrt{\log(\frac{KL\pi^2 t^2}{\delta})}$, where $c \in \mathbb{R}^+$. We test different values of c in experiments.

VI. EXPERIMENTS

We test our UCB-Replanning algorithm (*UCB*) on the application of aircraft flight path planning with partially observed wind. We compare our algorithm to three baselines: (1) receding horizon based Oracle (*Oracle*), (2) receding horizon based Plan by Mean (*Mean*) and (3) Great Circle Rout (*GCR*), namely following the shortest path in geodesic

²We experimentally verified that incorporating wind estimation into the computation of t_g significantly worsen the performance. This is because the wind estimation around the area that is far away from the airplane's current position is usually low-quality.

	Tail Wind	Head Wind
Oracle	18.8%	88.2%
UCB	5.4%	84.0%
Mean	2.9%	75.2%

TABLE I: Percentage improvement of Oracle, UCB and Mean compared to simply traveling along the straight line, under the synthetic wind filed setup. The percentage is computed as the difference between traveling time on straight line and traveling time of UCB (Oracle, Mean) divided by the traveling time on straight line.

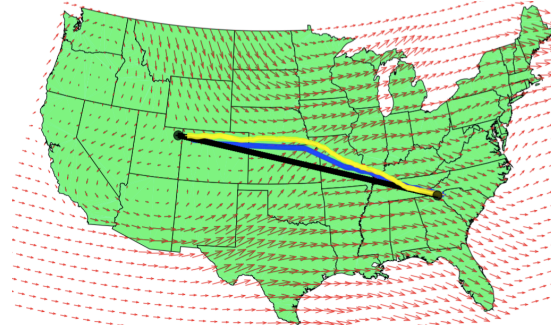
distance. The *Oracle*, which has access to the true wind information (i.e., it knows $g(\mathbf{x})$), picks trajectory using wind map $g(\cdot)$. Namely *Oracle* replace b_k in Alg. 1 using $f_{t,k}(g(\mathbf{x}_{t,0}^k), \dots, g(\mathbf{x}_{t,L}^k))$. *Mean* doesn't have access to the true wind information. Instead, *Mean* exactly follows the same structure of UCB-Replanning, but replace b_k in Line 6 of Alg. 1 using $f_{t,k}(\mu_{t-1}(\mathbf{x}_{t,0}^k), \dots, \mu_{t-1}(\mathbf{x}_{t,L}^k))$ (i.e., use the mean μ_{t-1} but ignore the standard deviations σ_{t-1}).

A. Synthetic Wind Field

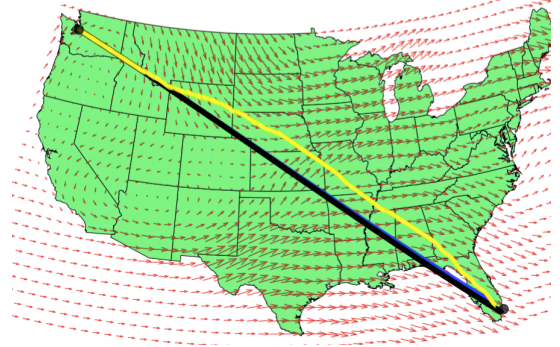
We created a wind field as shown in Fig. 3. The arrow indicates the direction of the wind filed and the length of the arrows indicates the strength of the wind. We pre-computed 25 trajectories, where each trajectory is simply a straight line with 30 segments. We set the length of each segment to be 0.2, the norm of the airplane's speed to 2.0, and the maximum norm of wind speed to 1.0.

When the airplane is traveling downwind, the strategy is to leverage the stronger wind to shorten the traveling time, as show by the trajectory of Oracle (blue) and the trajectory of UCB (red) in Fig. 3 (a). When the airplane is traveling upwind, one strategy to save traveling time is to identify the regions where the wind is not strong, which is exactly what Oracle, UCB and Mean performed in Fig. 3 (b). Also as we can see from Fig. 3, UCB's trajectory and Oracle's trajectory are usually different due to possible exploration at the beginning, but then gradually converges to each other. On the other hand, Mean may perform quite sub-optimally as shown in Fig. 3 (a).

Table I shows the percentage improvement of Oracle, UCB and Mean over GCR. The data used to compute the numbers in Table I is collected from 100 trials with different wind speed, and start/goal positions. As we can see, UCB consistently outperforms Mean, especially in the head wind case. Oracle generally performs the best since it has access the true underlying wind field (not available in practice). In summary, the comparison clearly shows that the tradeoff in exploration and exploitation introduced by UCB strategy is beneficial for Receding Horizon Control, while pure exploitation based strategy (i.e., Mean) in some cases can perform sub-optimally.



(a) South Carolina to Utah (Head Wind)



(b) Seattle to Miami (Tail Wind)

Fig. 4: Examples of trajectories resulting from Mean (Yellow) and UCB (Blue) for a short rout from South Caroline to Utah (a), and a long rout from Seattle to Miami (b). We also plot the Great Circle Rout (black).

B. Real Wind Field

We also tested our algorithm on wind map constructed from real data. We define boundaries to be the Continental United States. The Northwest boundary was set as (49.5N, 125.0W) and the Southeast as (25.0N, 67.5W). The simulated aircraft, maintains a constant cruising speed of 250 knots at an altitude of 39000 feet (11887 m). Winds encountered at this altitude could go upto upwards of 100 knots. Using realistic data provided by NOAA we construct wind maps by fitting a Gaussian Process over wind data from all 176 stations that NOAA maintains (e.g., Fig. 4 shows one instance of generated routs). Since it is impossible to get true wind map over US, we simply use the mean of the fitted GP as an estimation of ground truth of the wind speed. We refer readers to [25] for details of wind map construction.

We use an existing pre-computed library of trajectories from [1]. We test UCB and Mean on two different routes: (1) a *short* route from South Carolina to Utah (around 1300 nautical miles), and (2) a *long* route from Seattle to Miami (around 2700 nautical miles). As we can see from Fig. 4 (a), when flying with head wind, both UCB (and Mean in this case) exhibits another strategy to save traveling time: it guides the aircraft to fly in the direction that is nearly perpendicular to the wind speed in order to cancel the wind effect when the wind is strong. When flying with good tail wind as shown in Fig. 4 (b)), UCB almost identifies the

	South Carolina to Utah	Seattle to Miami
UCB	21079.7 ±1109.0	31333.1 ±1269.0
Mean	21183.3±1263.1	31716.5±1016.0
GCR	33712.5±1852.1	48195.7±1952.7

TABLE II: Average traveling time (seconds) with standard deviation resulting from UCB, Mean and GCR under the real wind field setup.

shortest path (Great Circle Rout) and follow it. Note that the trajectory resulting from UCB shown in Fig. 4 (b) is still a little bit different from GCR.

We simulate 11 days of real wind data by dividing each day into 6 hour time slots and simulating both paths for every slot (Fig. 4 shows one instance of the constructed wind maps). This in total give us 80 different trials for UCB and Mean, 40 for head wind and 40 for tail wind. We report the average traveling time and standard deviation in Table II. As we can see, in average UCB outperforms, and both UCB and Mean significantly outperform GCR.

We tested a variant of UCB and Mean, where we incorporated the estimation of wind speed to compute the time to goal (t_g) as shown in Eqn. 14. Due to the low quality estimation of wind speed at the areas far away from the aircraft's current position, using wind estimation to compute t_g actually worsen the performance of both UCB and Mean.

VII. CONCLUSION

We present UCB-Replanning, an online receding horizon based path planner that operates in an environment with latent information that can be modeled by Gaussian Processes. Equipped with a pre-computed trajectory library, at every iteration UCB-Replanning algorithm picks a trajectory to execute while collecting observations of the latent information on the fly to update the Gaussian Process. UCB-Replanning leverages the idea of optimism in the face of uncertainty to tradeoff exploration and exploitation in a near-optimal manner and achieve no-regret property with respect to an optimal decision maker that has full access to the latent information of the environment.

APPENDIX

A. Proof of Lemma 1

Proof: The proof is essentially the same as Lemma 5.1 in [21]. For completeness we present the proof here. Fix t and $\mathbf{x} \in D_t$. Note that under our assumption that f is a sample from the prior of GP, we have $g(\mathbf{x}) \sim \mathcal{N}(\mu_{t-1}(\mathbf{x}), \sigma_{t-1}(\mathbf{x})^2)$. Hence we have $(g(\mathbf{x}) - \mu_{t-1}(\mathbf{x}))/\sigma_{t-1}(\mathbf{x}) \sim \mathcal{N}(0, 1)$. The proof of Lemma 5.1 in [21] shows that if $r \sim \mathcal{N}(0, 1)$, we have $P(|r| \geq c) \leq \exp(-c^2/2), \forall c > 0$. This gives us the following result:

$$Pr(|g(\mathbf{x}) - \mu_{t-1}(\mathbf{x})| \leq \beta_t^{1/2} \sigma_{t-1}(\mathbf{x})) \geq 1 - \exp(-\beta_t/2).$$

We choose any sequence π_t such that $\sum \frac{1}{\pi_t} = 1$. For instance we can set $\pi_t = \pi^2 t^2 / 6$. Now let us set $\exp(-\beta_t/2) =$

$\frac{\delta}{\pi_t |D_t|}$, namely we set $\beta_t = 2 \log(\frac{|D_t| \pi_t}{\delta})$. Now use union bound over all rounds from 1 to T and over all LK waypoints in D_t , we can prove the above lemma. ■

B. Proof of Theorem 3

Proof: Let us define event B as

$$\begin{aligned} & \forall k \in [K], \forall t, |f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L) - f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L)| \\ & \leq l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^k), \end{aligned}$$

and from Lemma. 2 we know that the probability of event B happens is at least $1 - \delta$. Below we show that event B implies Theorem. 3. For the rest of the proof, we assume we condition on that event B happens. Consider round t . Note that I_t is defined as:

$$\begin{aligned} I_t &= \arg \max_{k \in [K]} f_{t,k}(\{\mu_{t-1}(\mathbf{x}_{t,j}^k)\}_{j=0}^L) \\ &+ l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^k), \end{aligned} \quad (15)$$

and I_t^* is defined as:

$$I_t^* = \arg \max_{k \in [K]} f_{t,k}(\{g(\mathbf{x}_{t,j}^k)\}_{j=0}^L), \quad (16)$$

namely the best trajectory one would pick at this round t if g is known. Now let us define the single step regret r_t as:

$$r_t = f_{t,I_t^*}(\{g(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L) - f_{t,I_t}(\{g(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L), \quad (17)$$

namely the regret one has by choosing I_t instead of I_t^* at round t . We can upper bound r_t using Eqn. 15 and 16 as follows:

$$\begin{aligned} r_t &= f_{t,I_t^*}(\{g(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L) - f_{t,I_t}(\{g(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) \\ &\leq f_{t,I_t^*}(\{\mu_{t-1}(\mathbf{x}_{t,j}^{I_t^*})\}_{j=0}^L) + l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t^*}) \\ &\quad - (f_{t,I_t}(\{\mu_{t-1}(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) - l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})) \\ &\quad \text{(event } B \text{ happens)} \\ &\leq f_{t,I_t}(\{\mu_{t-1}(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) + l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t}) \\ &\quad - (f_{t,I_t}(\{\mu_{t-1}(\mathbf{x}_{t,j}^{I_t})\}_{j=0}^L) - l\beta_t^{1/2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})) \\ &\quad \text{(Definition of } I_t \text{ from Eqn. 15)} \\ &= 2l\beta_t^{1/2} \sum_{j=0}^{L-1} \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t}). \end{aligned} \quad (18)$$

The square of r_t can be bounded as follows:

$$\begin{aligned}
r_t^2 &= 4l^2\beta_t \left(\sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t}) \right)^2 \leq 4l^2\beta_t L \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2 \\
&\leq 4l^2\beta_T L \sigma^{-2} \sum_{j=0}^L \sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2 \sigma^2 \\
&= 4l^2\beta_T L \sigma^2 \sum_{j=0}^L \left[\frac{\sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2}{\log(1 + \sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2)} \log(1 + \sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2) \right] \\
&\leq 4l^2\beta_T L \sigma^2 \frac{\sigma^{-2}}{\log(1 + \sigma^{-2})} \sum_{j=0}^L \log(1 + \sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2) \\
&= 4l^2\beta_T L \sigma^2 C_1 \sum_{j=0}^{L-1} \log(1 + \sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2) \quad (19)
\end{aligned}$$

where $C_1 = \sigma^{-2}/\log(1 + \sigma^{-2}) \geq 1$ and the third inequality comes from the fact that the function $x/\log(1 + x)$ is non-decreasing when $x > 0$, and $\sigma^{-2}\sigma_{t-1}^2 \leq \sigma^{-2}$ because we assume $\sigma_{t-1}(\mathbf{x})^2 \leq \kappa(\mathbf{x}, \mathbf{x}) \leq 1$ for any \mathbf{x} .

Since the regret $\mathbf{R}_T = \sum_{t=1}^T r_t$, we must have $\mathbf{R}_T^2 \leq T \sum_{t=1}^T r_t^2$. Using Lemma 5.3 and Lemma 5.4 from [21], we can link \mathbf{R}_T to the maximum information gain as follows:

$$\begin{aligned}
\mathbf{R}_T^2 &\leq 4l^2\beta_T L^2 \sigma^2 C_1 T \sum_{t=1}^T \sum_{j=0}^L \log(1 + \sigma^{-2}\sigma_{t-1}(\mathbf{x}_{t,j}^{I_t})^2) \\
&\leq 4l^2\beta_T L^2 \sigma^2 C_1 T \gamma_T,
\end{aligned}$$

where γ_T is the maximum information gain defined as

$$\begin{aligned}
\gamma_T &= \max_{A \subseteq \mathbb{X}, |A|=LT} I(v_A; g) \\
&= \max_{A \subseteq \mathbb{X}, |A|=LT} H(v_A) - H(v_A|g),
\end{aligned}$$

where $H(x)$ is the entropy of the random variable x , $H(x|y)$ is the conditional entropy, $v_A = \{g(\mathbf{x}) + \epsilon\}_{\mathbf{x} \in A}$ is the set of observations of $g(\mathbf{x})$ for all states \mathbf{x} in set A . Namely γ_T quantifies the maximum reduction in uncertainty about g from revealing the observations of g on LT states.

Theorem 5 from [21] shows that $\gamma_T \leq O(d \log LT)$ when $\kappa(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$ and $\gamma_T \leq O((\log LT)^{d+1})$ when $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-c\|\mathbf{x} - \mathbf{x}'\|^2)$. Substitute these results to the above inequality, we prove the theorem. ■

REFERENCES

- [1] C. Green and A. Kelly, "Toward optimal sampling in the space of paths," in *13th International Symposium of Robotics Research*, 2007.
- [2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, et al., "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.
- [3] D. Dey, K. S. Shankar, S. Zeng, R. Mehta, M. T. Agcayazi, C. Eriksen, S. Daftry, M. Hebert, and J. A. Bagnell, "Vision and learning for deliberative monocular cluttered flight," in *Field and Service Robotics*. Springer, 2015, pp. 391–409.
- [4] S. Arora, S. Choudhury, D. Althoff, and S. Scherer, "Emergency maneuver library-ensuring safe navigation in partially known environments," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 6431–6438.

- [5] C. Papadimitriou and J. N. Tsitsiklis, "The complexity of markov decision processes," *Math. Oper. Res.*, vol. 12, no. 3, pp. 441–450, Aug. 1987. [Online]. Available: <http://dx.doi.org/10.1287/moor.12.3.441>
- [6] J. Pineau, G. Gordon, S. Thrun, et al., "Point-based value iteration: An anytime algorithm for pomdps," in *IJCAI*, vol. 3, 2003, pp. 1025–1032.
- [7] H. Kurniawati, D. Hsu, and W. S. Lee, "Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces," in *RSS*, vol. 2008. Zurich, Switzerland, 2008.
- [8] T. Smith and R. Simmons, "Heuristic search value iteration for pomdps," in *UAI*. AUAI Press, 2004, pp. 520–527.
- [9] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 2164–2172. [Online]. Available: <http://papers.nips.cc/paper/4031-monte-carlo-planning-in-large-pomdps.pdf>
- [10] B. Bonet and H. Geffner, "Solving POMDPs: RTDP-Bel vs. point-based algorithms," in *IJCAI*, 2009.
- [11] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The International Journal of Robotics Research*, 2009.
- [12] R. Platt Jr et al., "Belief space planning assuming maximum likelihood observations," in *Proceedings of the Robotics: Science and Systems Conference*, 6th, 2010.
- [13] J. Van Den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *IJRR*, vol. 31, no. 11, pp. 1263–1278, 2012.
- [14] S. Patil, G. Kahn, M. Laskey, J. Schulman, K. Goldberg, and P. Abbeel, "Scaling up gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation," in *Algorithmic Foundations of Robotics XI*. Springer, 2015, pp. 515–533.
- [15] W. Sun, J. Van Den Berg, and R. Alterovitz, "Stochastic extended lqr: Optimization-based motion planning under uncertainty," in *Algorithmic Foundations of Robotics XI*. Springer, 2015, pp. 609–626.
- [16] W. Sun, S. Patil, and R. Alterovitz, "High-frequency replanning under uncertainty using parallel sampling-based motion planning," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 104–116, 2015.
- [17] D. Dey, A. Kolobov, R. Caruana, E. Kamar, E. Horvitz, and A. Kapoor, "Gauss meets canadian traveler: shortest-path problems with correlated natural dynamics," in *AAMAS*, 2014, pp. 1101–1108.
- [18] A. Olsen, "Pond-hindsight: Applying hindsight optimization to partially-observable markov decision processes," Master's thesis, Utah State University, 2011.
- [19] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002.
- [20] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The non-stochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002.
- [21] N. Srinivas, A. Krause, S. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," in *ICML*. Omnipress, 2010.
- [22] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Foundations and Trends® in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.
- [23] R. I. Brafman and M. Tennenholtz, "R-max-a general polynomial time algorithm for near-optimal reinforcement learning," *Journal of Machine Learning Research*, vol. 3, no. Oct, pp. 213–231, 2002.
- [24] L. Li, "Sample complexity bounds of exploration," in *Reinforcement Learning*. Springer, 2012, pp. 175–204.
- [25] A. Kapoor, Z. Horvitz, S. Laube, and E. Horvitz, "Airplanes aloft as a sensor network for wind forecasting," in *Proceedings of the 13th international symposium on Information processing in sensor networks*. IEEE Press, 2014, pp. 25–34.