Gauss Meets Canadian Traveler: Shortest-Path Problems with Correlated Natural Dynamics

Debadeepta Dey[†], Andrey Kolobov[‡], Rich Caruana[‡]
Ece Kamar[‡], Eric Horvitz[‡], Ashish Kapoor[‡]
[†]Carnegie Mellon University, Pittsburgh, PA 15213 USA
[‡]Microsoft Research, Redmond, WA 98052 USA

Categories and Subject Descriptors

I.2 [**Problem Solving, Control Methods, and Search**]: Plan execution, formation, generation

Keywords

Planning; Canadian Traveler Problem; Gaussian Process; Aircraft Routing

ABSTRACT

In a variety of real world problems from robot navigation to logistics, agents face the challenge of path optimization on a graph with unknown edge costs. These settings can be generally formalized as the Canadian Traveler Problems (CTPs) [**?**]. Although in many applications the edge costs have dependencies resulting from world dynamics, CTPs with such structure have received considerably less attention than those with independent edge costs, largely because the dependence structure is often problem-specific and difficult to state compactly. Yet, in a wide variety of navigation tasks, spatial correlations between edge traversal costs are governed by natural phenomena such as winds, congestion, or ocean currents, which are conveniently described with a well-understood machine learning model — Gaussian Process (GP). In this article, we propose a synthesis of CTPs and GPs, the Gaussian Traveler Problem (GTP). In GTPs, an agent observes the costs of graph edges when traversing them, and uses the observed costs to adjust its belief over other edges via Gaussian Process updates. Examples of GTP instances include aircraft, traffic, and vessel navigation, to name just a few. Computing optimal agent behavior for a GTP turns out to be equivalent to solving a Partially Observable MDP with continuous observation space. We present an approximate algorithm for solving GTPs with efficient machine-learning and decision-making components, whose design is influenced by the challenges of real-world problems. Despite the intractability of computing an optimal policy, our experiments in the aircraft navigation scenario with real wind data demonstrate that our framework can significantly improve upon state-of-the-art techniques for planning airplane routes.

The project was formulated and conducted during an internship by Debadeepta Dey at Microsoft Research.

Appears in: *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), Lomuscio, Scerri, Bazzan, Huhns (eds.), May, 5–9, 2014, Paris, France.*

1. INTRODUCTION

An important problem faced by many agents acting in the physical world is choosing an optimal route in a graph from a source to a destination in the absence of exact knowledge about the graph's edge costs. As one of many examples, consider a Mars rover attempting to reach the intended location of a scientific experiment or generally a robot trying to move to a designated area over unexplored terrain. The robot may have very rough information about the terrain's treacherous regions that would risk slow it down, and needs to travel to its objective using only this partial data and the observations it makes along the way. Scenarios of this kind can be formalized as *Canadian Traveler Problems* (CTPs) [**?**], also known as *stochastic shortest path problems with recourse* [**?**]. In a CTP, an agent learns about the edge costs as it is exploring a graph, and dynamically adapts its plans based on the newly revealed information. Most work on CTPs has focused on the subclass of these problems where edge costs are assumed to be independent; upon arriving at a node, an agent observes the costs of adjacent edges (such as the firmness of the terrain in a neighborhood of a location in the above example) but receives no information about edges elsewhere in the graph. However, in many real domains, edge costs *are* statistically dependent; for example, terrain traversability over sizable regions of the map tends to be similar. Nonetheless, CTPs with edge cost dependencies have received far less attention in the literature, largely due to the domain-specific, cumbersome form of the dependencies that complicates the design and analysis of general algorithms for these problems.

In this paper, we address this need by presenting an expressive model that can characterize the dependency structure in a broad class of settings. Our approach makes use of the observation that in many scenarios involving natural phenomenon, edge costs are influenced by the *natural dynamics* of the world (e.g., physical forces, weather, etc.) and are dependent only via such natural dynamics. In these domains, modeling the global natural phenomena is the key to describing edge cost interrelations. *Gaussian Processes* (GPs) [**?**] is a machine learning framework that fits this task well. Many of GPs' successes have come from modeling spatially correlated natural patterns including winds [**?**], oceanic currents [**?**], and traffic volume [**?**]. GPs offer an intuitive representation of the dependencies in natural dynamics and enable efficient prediction of the natural dynamics realizations (e.g., wind vectors) at any location, conditioned on observations at other locations. GPs can also be easily updated with new observations, continually providing an agent with an accurate idea of what to expect based on the available data.

In this article, we couple CTPs with GPs and investigate solutions for the resulting formalism on the example of wind-aware aircraft navigation. Imagine an aircraft whose pilot is free to choose any trajectory to a destination and wishes to minimize flight time. Pilots typically fly their aircraft at a constant recommended airspeed (speed w.r.t. the surrounding air) for most of the flight [**?**]. At a fixed airspeed and a given altitude, an aircraft's flying time depends almost exclusively on the chosen flight path and encountered winds. Thus, constructing an optimal flight trajectory on an appropriately discretized map would be a simple planning problem if the pilot had access to detailed wind information over the entire map. Unfortunately, pilots today usually commit to a flight plan based on the very scarce wind data available at the beginning of the flight. Modeling wind correlations with GPs allows an aircraft's onboard systems to predict winds in other regions from the local winds in the aircraft's vicinity and to alter the route accordingly. This turns the flight planning task into a CTP with edge costs represented by a GP.

More formally, we propose a new framework, called the *Gaussian Traveler Problem* (GTP), to model scenarios such as aircraft, vessel, and traffic navigation. GTPs unite the strengths of CTPs with GPs. In these problems, edge costs are functions of a natural dynamics characterized by a GP (e.g., in the aircraft example, edge costs are traversal times, which are a function of the wind at different locations). As we demonstrate, every GTP maps to a goal POMDP [**?**] with a continuous observation space. The belief state of such a POMDP consists of the agent's geographic location (known exactly) and the agent's belief about the costs of every edge in the graph, induced by the current parameters of the GP. Belief tracking in this POMDP amounts to performing a GP update using an observed edge cost. The POMDP's goal is any state where the agent's geographic location matches its destination.

Continuous observation spaces and the sheer size of the underlying graph make the optimal solution of optimally GTP intractable for realistic scenarios. We present approximate online solution algorithms that combine reasoning about uncertainty with existing deterministic planning methods to make decisions in a tractable way. Using the example of the aircraft navigation problem, we demonstrate that solutions computed with these sampling-based replanning algorithms significantly outperform the industry standard on real-world wind data. We make the following contributions:

- We formulate a new type of CTP with edge cost dependencies, the Gaussian Traveler Problems, that integrate the modeling power of Gaussian Processes into the CTP framework.
- We cast a realistic aircraft routing example scenario as a GTP.
- We describe a procedure for converting GTPs to equivalent POMDPs and show, based on publicly available historical wind data for the continental US, that efficient, sampling-based approximate algorithms yield 5-10% shorter flying times than the current state-of-the-art approach in the aircraft navigation domain.

2. BACKGROUND AND RELATED WORK

This paper relates to three main areas of prior work: Canadian Traveler Problems, Gaussian Processes, and POMDPs. Before describing the paper's contributions in more detail, we review relevant material from these areas.

Canadian Traveler Problems. Canadian Traveler Problems (CTPs) were originally inspired by the difficulties of planning routes for trucks in Canada, where road segments can become snowed over and impassable unbeknownst to truck drivers, and have been studied in many variants [**?**, **?**, **?**]. In general, CTPs are concerned with getting from a source node v_0 to a goal node v_g in a graph $G = \langle V, E \rangle$, whose edges have unknown costs and/or traversability. Since an edge's impassability can be modeled by assigning the edge a very high cost, we will discuss CTPs only from the standpoint of their cost structure.

In most CTP formulations, in order to optimize the cost of reaching v_g from v_0, an agent needs to reason about the *joint edge cost realizations* possible for the given problem. A joint edge cost realization $C : E \to \mathbb{R}^+$ is an assignment of costs values to all of the graph's edges. The agent knows the set $\mathcal{R}$ of all possible joint cost realizations before it starts acting. In addition, it knows a prior probability distribution P over the set $\mathcal{R}$. P allows the agent to hypothesize apriori about the costs of different edges. However, with the notable exception of *CTPs with remote sensing* [**?**], in which an agent can query costs of remote edges, an agent can find out an edge's actual cost only once it is at one of the edge's endpoints or attempts to traverse it.

Based on the observed edge cost, the agent updates the distribution P by eliminating the joint cost assignments that do not agree with the acquired observation. Thus, a CTP can be formalized as a tuple $\langle V, E, v_0, v_g, P_0, \mathcal{R} \rangle$ of the above components. Solving a CTP entails computing a policy to reach the destination while incurring the least expected or worst-case cost. Note that CTPs naturally lend themselves to adaptive solutions: an agent can choose a path to follow from v_0 based on the knowledge of incident edge costs and the distribution P, follow the first edge in the path, arrive at another node v_1, learn the costs of surrounding edges, update P and alter its initial plan accordingly, and so on.

Although the described formulation lets edges have arbitrary cost dependencies, most prior work has assumed edge costs to be independent, since this allows the set $\mathcal{R}$ of *joint edge cost realizations* to be compactly described as a Cartesian product of individual edges' possible cost sets, and allows the joint distribution P to be factored into the product of cost distributions for each edge. In contrast, if edge cost distributions are dependent, in general the distribution P needs to be specified as an explicit list of joint cost realizations $\mathcal{R}$ with associated probabilities [**?**], which grows exponentially with the number of edges in the graph. This representation quickly becomes infeasible for realistic scenarios, making algorithms for general CTPs inefficient. In this paper, we propose an alternative, closed-form representation for the joint edge-cost prior P that is suitable for a number of practical domains that involve natural dynamics. We discuss this representation, Gaussian Processes, next.

Gaussian Processes. At the highest level of abstraction, a Gaussian Process (GP) [**?**] is a distribution over functions that respect certain smoothness constraints and agree well with the observed data. GPs have been previously used to model various spatially correlated natural phenomena [**?**, **?**, **?**]. In this paper, we propose to employ them in CTP scenarios where such natural phenomena govern the world dynamics and ultimately influence edge costs. In particular, using wind as our running example, we can view a wind

pattern over a territory as a function mapping geographic coordinates to wind vectors. We may not know the wind pattern over the entire map, but with the help of GPs we can easily express the fact that winds tend to be similar in nearby regions and represent a distribution of our hypotheses about winds at unobserved locations.

Our main reason for using GPs is that they provide an appealing probabilistic framework where the uncertainty in the value of the function output (e.g., wind uncertainty on graph edges that have not been visited yet) is modeled conditioned on the observations (of winds on edges that the agent has traversed). GPs consider the observed wind vector t for a location $\mathbf{x}$ to be a noisy version of a latent variable y. In the wind example, y can be the true predominant wind strength and direction at location $\mathbf{x}$ during a given time period, and t, a noisy vector of the wind that is observed directly. Specifically, the relationship between y and the observation t is characterized by a Gaussian likelihood model.

$$p(t|y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(t-y)^2}{2\sigma^2}}, \quad (1)$$

where σ^2 is the noise model variance. Thus, GPs assume the observations to be generated from hidden variables by corrupting these variables' values with a 0-mean Gaussian noise.

Crucially, GP imposes a smoothness constraint on the latent variables' values by defining a probabilistic relationship between any finite set of locations $\mathbf{X} = \{\mathbf{x}_1, .., \mathbf{x}_n\}$ and the corresponding latent variables $\mathbf{Y} = \{y_1, ..., y_n\}$ s.t. the distribution $p(\mathbf{Y}|\mathbf{X})$ gives higher probability to wind vectors that respect some notion of similarity between locations. Intuitively, GPs assume that similar locations should have the same wind; to reflect this, the similarity between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ is defined via a *kernel function* $k(\mathbf{x}_i, \mathbf{x}_j)$. Using a kernel function k, GPs implement the smoothness constraint by letting $p(\mathbf{Y}|\mathbf{X})$ be a Gaussian, i.e., $p(\mathbf{Y}|\mathbf{X}) \sim \mathcal{N}(0, \mathbf{K})$, where $\mathbf{K}$ is an $n \times n$ *kernel matrix*.

Now, suppose we have wind readings $\mathbf{t}_L = \{t_1, \ldots, t_m\}$ for some of the locations $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbf{X}$, and would like to update our hypothesis about wind patterns accordingly. GP makes computing the hypothesis posterior, $p(\mathbf{Y}|\mathbf{X}, \mathbf{t}_L)$, easy by combining the smoothness constraints $p(\mathbf{Y}|\mathbf{X})$ imposed via the GP prior and the information provided by the observations via $p(\mathbf{t}_L|\mathbf{Y})$.

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{t}_L) \propto p(\mathbf{Y}|\mathbf{X}) p(\mathbf{t}_L|\mathbf{Y}) = p(\mathbf{Y}|\mathbf{X}) \prod_{i=1}^{m} p(t_i|y_i). \quad (2)$$

As this equation shows, the posterior is just a product of Gaussians and hence is a Gaussian itself. For each unobserved location $\mathbf{x}_u$, the parameters of this Gaussian are given by [?]

$$\bar{y}_u = \mathbf{k}(\mathbf{x}_u)^T (\sigma^2 \mathbf{I} + \mathbf{K}_{LL})^{-1} \mathbf{t}_L \quad (3)$$

$$\Sigma_u = k(\mathbf{x}_u, \mathbf{x}_u) - \mathbf{k}(\mathbf{x}_u)^T (\sigma^2 \mathbf{I} + \mathbf{K}_{LL})^{-1} \mathbf{k}(\mathbf{x}_u). \quad (4)$$

Here, $\mathbf{k}(\mathbf{x}_u)$ is the vector of kernel function evaluations with the m observed locations in $\mathbf{X}$, $\mathbf{K}_{LL} = \{k(\mathbf{x}_i, \mathbf{x}_j)\}$ is the $m \times m$ data covariance over the observed locations, and σ^2 is the noise variance. Moreover, for every unobserved location, we can also compute a posterior for the observations, which is also Gaussian and has the form $p(t_u|\mathbf{X}, \mathbf{t}_L) \sim \mathcal{N}(\bar{y}_u, \Sigma_u + \sigma^2)$ [?].

The assumption of zero mean is for clarity only; GPs trivially handle non-zero-mean beliefs.

The performance of GP-based prediction depends strongly on the chosen kernel function k and its parameters. There are many ways to learn these parameters from prior data; we use evidence maximization by performing gradient descent [?, ?] for this purpose.

POMDPs. The CTP versions in which the agent gradually collects information about edge costs as it is navigating the graph are known to be equivalent to instances of Partially Observable Markov Decision Processes (POMDPs). Generally, POMDPs are a framework for describing planning scenarios where the agent's objective is to maximize reward or minimize incurred cost by choosing actions based on partial information about the current world state; in the case of CTPs, the objective function is the expected cost of getting from the start to the destination node in a given graph. Formally, the *goal-oriented POMDPs* [?] that describe such settings are tuples $\langle S, A, T, C, G, Z, O, b_0 \rangle$, where S is a set of world states, A is a set of actions the agent can perform, $T : S \times A \times S \to [0, 1]$ is a transition function that gives the probability of transitioning from a state s to a state s' when the agent executes action a, $C : S \times A \times S \to \mathbb{R}^+$ is a cost function specifying the cost the agent pays when transitioning from s to s' using a, $G \subseteq S$ is a set of goal states the agent is attempting to reach, Z is a set of possible observations indicative of the current world state, $O : S \times A \times Z \to [0, 1]$ is an observation function giving the probability of getting a particular observation z when the agent executes action a and ends up in state s, and b_0 is a distribution over world states characterizing the agent's initial belief about the state where the agent starts. The agent has no direct knowledge of world states. It can only infer a belief about them by analyzing observations it gets when it executes various actions. Solving a POMDP means finding a (Markovian deterministic) *policy* $\pi : B \to A$ that recommends actions based on the agent's current belief.

The CTP variant we propose in this paper can be converted to a goal-oriented POMDP as well. We describe the conversion process in the next section, and in the meantime point out the main purpose of performing such a conversion is to make CTPs amenable to the vast amount of machinery that has been developed for solving POMDPs.

An optimal solution for a POMDP can be computed with dynamic programming, but doing so is PSPACE-complete [?]. To address this intractability, researchers have proposed various approximate solution methods, e.g., the approaches derived from point-based value iteration [?, ?] and Monte-Carlo planning [?, ?]. However, these algorithms focus on goal-free reward-oriented POMDPs and are not directly applicable to the problems we consider here. Techniques targeting goal-oriented POMDPs (e.g., [?]) are more relevant, but Gaussian Traveler Problems studied in this paper have a number of challenges making them difficult even for these specialized methods. These challenges include continuous observation space, large number of time steps needed to reach the goal, and computational limitations imposed by the real-world problem specifications. In this paper, we apply determinization to offer tractable approximate solutions to GTPs. Similar ideas have been applied for solving POMDPs, e.g., in [?].

3. PROBLEM FORMULATION

Our primary motivation comes from the insight that in many CTP scenarios, including the aircraft navigation example studied in this paper, the only source of non-deterministic

dependencies among costs of edges are the *natural dynamics* governed by phenomena such as winds. We extend the notion of natural dynamics from its original meaning in robotics [**?**] to refer to any phenomenon that interferes with an agent's actions and that it cannot control. For a broad range of problems, their natural dynamics can be successfully described by a Gaussian Process, a model easy both to formulate and to update. Thus, although stochastic relationships among edge costs in general CTPs are hard to characterize concisely, GPs address this issue in many interesting and practically important cases. In this section, we present a novel class of CTPs that exploits this characteristic by having GPs as an inherent component of the problem definition.

Letting $\mathrm{GP}(0, k, \sigma^2)$ denote a Gaussian Process with zero mean, kernel function k, and observation noise variance σ^2, we can define our new CTP subclass as follows:

Definition. ***Gaussian Traveler Problem.*** *A Gaussian Traveler Problem (GTP) is a tuple* $\langle V, E, v_0, v_g, \mathcal{W}, k, \sigma^2, C\rangle$*, where*

- V*,* E*,* v_0*, and* v_g *are as in the CTP definition;*
- $\mathcal{W}$ *is a set of* joint natural dynamics realizations $W : E \to \mathbb{R}^n$*, each of which maps all edges to the values of natural dynamics on those edges (e.g., true wind magnitude and direction on the edges);*
- $k : E \times E \to \mathbb{R}$ *is a kernel function encoding the similarity between any two edges;*
- σ^2 *is the variance of the natural dynamics observation noise;*
- $C : (E \to \mathbb{R}^n) \to (\mathbb{R}^+)^{|E|}$ *is an invertible* dynamics-dependent cost function *that maps joint natural dynamics realizations to joint assignments of positive costs to all the edges;*

Solving a GTP means finding a policy π *with the least expected cost of leading the agent from* v_0 *to* v_g*, given that the prior distribution over joint edge cost realizations is* $C(\mathbf{W})$*, where* $\mathbf{W} \sim GP(0, k, \sigma^2)$ *is a random variable with domain* $\mathcal{W}$*.*

The GTP definition differs from that of CTP only in the formalization of the prior belief over joint edge cost realizations. In particular, GTP views an edge cost realization (e.g., edge traversal times) as being generated from a realization W (e.g., wind vectors in the vicinity of every edge), which, in turn, is generated by the GP-governed natural dynamics of the problem (the probabilistic model that describes wind correlations at different locations).

As in many existing CTP flavors, we assume that a particular natural dynamics realization (e.g., wind vectors on all edges), and hence a joint edge cost realization, has been "chosen" by nature by the time an agent starts acting and remains fixed until the agent reaches the goal. Thus, the agent only needs to "uncover" the chosen cost realization by observing costs on various edges (which are guaranteed to remain unchanged once observed) and updating the GP that describes the underlying natural phenomenon. Note, however, that in order to update the GP using an edge cost observation, an agent needs to infer the value of wind on that edge from the observed edge cost. The requirement of the cost function's invertibility in the GTP definition guarantees that this is possible.

Although GPs provide a convenient tool for representing beliefs about true edge costs, they don't provide a policy that tells an agent how to act depending on these beliefs. We formalize decision-making in GTPs as a goal POMDP and then adapt POMDP algorithms to compute approximate GTP solutions:

Theorem. *Every GTP can be converted into a goal POMDP with continuous observation space.*

Proof. The theorem holds because GTPs are a subclass of CTPs. However, we present an explicit GTP-to-POMDP conversion procedure because it shows how the properties inherent to GTPs translate to the properties of the resulting POMDPs. Given a GTP, the state of the corresponding POMDP at any given point is uniquely described by the combination of the graph node where the agent is at the moment and the world's joint natural dynamics realization (unavailable to the agent). The goal of the POMDP is any state in which the agent is at the graph's goal node. To get to that node, the agent travels along the graph's accessible edges, i.e., each accessible edge maps to a POMDP action. Crucially, by our assumption, the natural dynamics realization component of the POMDP state never changes — it remains the same in any state the agent visits from the state where the agent starts. The POMDP's cost function is fully determined by the joint natural dynamics realization implanted into the state. The POMDP's belief states, like its actual states, consist of two parts: the agent's current node, which is known to the agent exactly, and the agent's GP belief about the edge costs. Initially, the agent's belief is given by $\mathrm{GP}(0, k, \sigma^2)$. However, the agent can update it by receiving an observation (an edge traversal cost), inverting the GTP cost function to find the natural dynamics realization for that edge, and using Equations **??** and **??** to re-estimate the GP's covariance matrix and means for all other edges.

More formally, consider a GTP $\langle V, E, v_0, v_g, \mathcal{W}, k, C\rangle$. To build a corresponding POMDP, define the POMDP's state space S to be the set $V \times \mathcal{W}$, i.e., the cross product of graph node set and the set of all possible natural dynamics realization. The goal set G consists of all POMDP states whose agent location component equals v_g. Define the POMDP's action space A by creating an action for every edge $e \in E$ accessible from a given node. Since an agent's actions do not alter the world's natural dynamics realization in GTPs and merely cause the agent to travel from one node to the next, the POMDP's transition function T follows naturally as $T(s, a_e, s') = 1$ if $s = \langle v, W\rangle$ and $s' = \langle v', W'\rangle$ are s.t. $W = W'$ and action a_e corresponds to edge e connecting v and v' in the GTP. In all other cases, $T(s, a_e, s') = 0$. Similarly, the POMDP's cost function $C(s, a_e, s') = C_e(W)$ for any states $s = \langle v, W'\rangle$ and $s' = \langle v', W''\rangle$, where $C_e(W)$ denotes the cost of edge e under natural dynamics realization W, $W = W' = W''$, e connects v and v' in the GTP, and a_e is the POMDP's action corresponding to edge e. In all other cases, $C(s, a_e, s')$ is infinite or undefined. The set Z of the POMDP's possible observations equals $\mathbb{R}^+$, the set of all possible edge cost values. This is because once an agent executes an action (traverses an edge), the only observation it gets is the cost of that edge. The POMDP's observation function O is defined as $O(s, a_e, z) = 1$ if $z = C_e(W)$, where $s =< v, W >$ and action a_e corresponds to edge e, and 0 otherwise. Finally, the POMDP's initial belief state is $\langle v_0, GP(0, k, \sigma^2)\rangle$. In general, the POMDP's belief space

consists of all pairs whose first component is a node in V and whose second component is a multivariate Gaussian in $|E|$ dimensions, with some dimensions possibly collapsed to Dirac delta functions.

By construction, there is a one-to-one mapping between the policies in the original GTP and the described POMDP that preserves the ranking of the policies in terms of their expected cost of reaching the goal [**?**]. Thus, a policy is optimal for the GTP if and only if its counterpart is optimal for the POMDP, completing the proof. ■

The continuous observation space and large state spaces of GTP instances arising from real-world scenarios render the existing POMDP solution algorithms infeasible for tacking GTP-derived POMDPs. We discuss these challenges and ways of circumventing them in Section **??**, but first give a detailed example of a real problem GTPs can model, aircraft navigation.

4. AIRCRAFT NAVIGATION EXAMPLE

A central challenge for each of the 30,000 flights taking off in U.S. every day is arriving to a goal location from the departure location with minimal time and fuel spent during the flight. Since aircraft primarily cruise at a predetermined constant speed relative to the surrounding air [**?**], the primary factor affecting the duration of a flight is wind patterns along the chosen route. Commercial aviation aircraft usually fly between the altitudes of 23000 to 41000 feet. In this altitude range, wind patterns can vary significantly in direction and in magnitude between 30 knots to 120 knots [**?**]. The true wind realizations (i.e., wind speeds and directions) are not known at the time of flight planning. Instead, the National Atmospheric and Atmospheric Administration (NOAA) of U.S. issues wind reports at 176 weather stations spread out over the country every 6 hours. These readings are known as the "Winds Aloft" data. They give wind speeds and magnitudes are obtained with weather balloons for altitudes up to 39000 feet over the mean sea level. Currently, aircraft routes are planned based on the Winds Aloft data at the time of take-off. Since there are only 176 stations for all of the U.S., this data is very sparse, and modern day route planners estimate wind at any location by linearly interpolating the winds at the nearest wind stations [**?**]. We term this approach *Linear Interpolation*.

To represent this navigation scenario as a GTP, we discretized the appropriate area of the map (e.g., the North American continent) into a regularly spaced grid (with nodes spaced 1 degree longitude and latitude apart) and connecting them to d of their nearest neighbors ($d = 8$ was used in our experiments). To compute the cost of traversing an edge e that connects nodes r miles away from each other for a fixed airspeed of l knots along the edge and the wind vector in the edge's vicinity being $\vec{w}$, we computed the projection w_e of $\vec{w}$ onto the vector representing the edge e, and let the edge cost be $C_e = \frac{r}{l - w_e}$. To avoid negative costs, we assumed that the wind magnitude never exceeds the aircraft's airspeed—an assumption that holds well in practice.

Before presenting a general algorithm for solving GTPs via converting them to goal POMDPs and using it for the aircraft navigation scenario, we explore simpler approaches to this special case. They will serve as baselines for our experiments in this domain. The first of them is directly based on the aforementioned *Linear Interpolation* technique. It uses the interpolation on the initial weather station readings

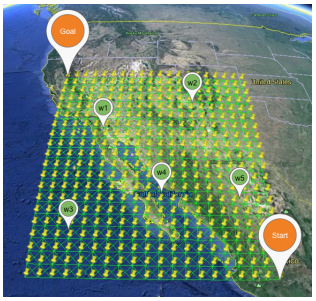

Figure 1: This 8-connected graph was used as the underlying graph for all experiments. The weather locations are marked with green markers and start and goal locations are marked with orange markers.

and a deterministic shortest-path algorithms such as A^* or Dijkstra's to construct a plan that is used without modifications throughout the flight. An alternative approach, which we call *No Replan*, is similar to the *Linear Interpolation* method, but differs in using the GP model to generate predictions about wind patterns. It trains the GP kernel with the available Winds Aloft reports, takes the GP's mean wind predictions to produce a cost (traversal time) estimate for every edge, and also uses A^* or Dijkstra's to find a path to follow for the whole flight.

Both the *Linear Interpolation* and *No Replan* algorithms fail to take into account all the wind measurements that the aircraft gathers as it traverses the edges on its way to the destination. The *Replan by Mean* scheme addresses this shortcoming by incorporating observations the aircraft acquires on its way to the goal. It keeps updating the GP every time a wind is measured by the aircraft, re-predicting the edge wind vectors in the graph, and replanning a new path from the current location to the goal. Although this approach can incorporate newly acquired observations into the planning process, it still suffers from a crucial deficiency: it fails to take into account the uncertainty of GP's wind predictions at every edge, reflected in GP's variance parameter. Our approach, described in the next section, rectifies this weakness.

5. ALGORITHMS FOR SOLVING GTPs

In Section **??**, we showed that every GTP can be converted to a goal POMDP with continuous observation space. The question is then: can we solve this problem with standard techniques from the POMDP literature? The optimal POMDP solution methods operate iteratively, where the complexity of every iteration depends on the number of possible observations in the problem [**?**]. Since in GTP POMDPs the number of observations is infinite, these algorithms cannot be used in their basic form to solve these problems. Point-based methods, the state of the art in solving POMDPs approximately, appear to be more feasible, but they, too, fail to scale to GTP POMDPs that model realistically-sized scenarios. The ultimate reason is the minimum number of edges an agent needs to traverse in order to get from v_0 to v_g in these settings, which can reach 200 for moderately-sized aircraft navigation settings. Performing belief state backups for this many steps is very intensive computationally, even if the issue of continuous belief state can be circumvented.

Instead, we take a determinization-based approach. Note

Figure 2: The wind pattern used in experiments presented in wind station randomization experiments.

that GPs that we are using to model the underlying natural dynamics reduce to multivariate Gaussian distributions for graphs with finite numbers of edges. Such a distribution can be easily sampled. Each sample from this Gaussian is a joint natural dynamics realization, i.e., a wind vector for every edge. Converting a sample's winds to costs (time of travel along the edge) for a fixed aircraft airspeed yields a deterministic planning problem on the same graph. By generating many such samples from the GP representing the current belief about the winds and solving them (an inexpensive operation, since deterministic planners are very efficient), we can can estimate the expected cost-to-goal for each available action choice at the aircraft's current node. We can then choose the action (edge) with the least expected cost-to-goal estimate, transition to another node via the corresponding edge, measure wind on that edge, update the GP, and repeat the process.

Algorithm **??** lays down the details of this approach, which we call *Replan by Sampling*. *Replan by Sampling* interleaves planning and execution. The inputs to the algorithm are the components of a GTP, including a GP with a kernel trained initially using the available weather stations, and the number of samples n to be generated at each step. At the initial node, the algorithm samples the GP n times to produce n separate natural dynamics realizations in the set $\{W\}$ (line **??**). Then, each available immediate action's (edge's) Q-value is estimated by running a deterministic shortest path algorithm on each sampled realization (lines **??**-**??**). The action with the least estimated cost-to-goal is chosen for execution (line **??**) in the real world. A new wind measurement z is obtained in line **??** and the GP is updated with the new observation in line **??** using Equations **??** and **??**. This process is repeated until the goal node v_g is reached. Due to its probabilistic nature, *Replan by Sampling* accounts for the uncertainty in wind prediction naturally modeled by the GP. Instead of following a fixed policy, this algorithm continuously replans throughout execution by incorporating the new wind observations the aircraft makes along the way.

Algorithm 1 `Replan by Sampling`

Require: v_o, v_g, V, E, k, C, $\texttt{GP}(0, k, \sigma^2)$, n
1: $v = v_o$
2: **while** $v \neq v_g$ **do**
3: $\{W\} = \texttt{Sample}(\text{GP}(0, k, \sigma^2), n)$
4: **for** each a in $\texttt{NeighboringEdges}(v)$ **do**
5: $Q(v, a) = \texttt{EstimateQ}(\{W\}, v, a)$
6: **end for**
7: $a^* = \texttt{argmin}_a\ Q(v, a)$
8: $v = \texttt{Traverse}(a^*, v)$
9: $z = \texttt{ReceiveNewObservation}(v)$
10: $\text{GP}(0, k, \sigma^2) = \texttt{UpdateGP}(GP(0, k, \sigma^2), z)$
11: **end while**

Algorithm 2 `EstimateQ`

Require: $\{W\}, v, a, v_g$
1: $Q = 0$
2: **for** each $C(W)$ in $\{W\}$ **do**
3: $v' = \texttt{Neighbor}(a, v)$
4: $q = \texttt{Dijkstra}(v', v_g, C(W))$
5: $Q = Q + q$
6: **end for**
7: return $Q = \frac{Q}{|\{W\}|} + c(v, v')$

6. EXPERIMENTS

We developed a simulation platform to evaluate the performance of the proposed method and to analyze the benefits gained from replanning approaches in various settings. The simulations enable us to understand how different factors, such as the number of weather stations available, their locations and wind characteristics, influence the efficiency of flight planning.

Our simulations are performed on a grid roughly corresponding to $\frac{1}{4}$ of the continental US, covering the US west coast and part of Mexico (Figure **??**). The grid nodes are spaced 1 degree of longitude and latitude apart, and each is connected to its 8 closest neighbors. In our experiments, one corner of the graph, Mexico City, is the start location and the opposite corner, Seattle, is the goal. The aircraft speed was set at 250 m/s (560 mph), representative of the cruising speed of a commercial jet airliner. Weather stations provide information about the wind patterns on the map before the plane takes off and collects observations. The number of weather stations available and their locations determine the amount of information available prior to a flight. In our simulations, we vary the number of weather stations between 5 and 50 and put them on the map randomly in different trials to observe their effect on planning.

In our simulations, we used a 2D polynomial equation to generate wind patterns on the map. As shown in previous work [**?**], wind patterns are primarily a function of temperature and pressure at high altitudes, and a quadratic polynomial gradient is a well-fitting representation of this dynamic. In our simulations, we vary the wind speed between 30 and 120 knots at the altitudes of 23000 and 41000 to agree with the observations of real wind speeds at altitudes where commercial aircraft fly [**?**]. By randomly perturbing the 2D polynomial's coefficients, we created wind patterns whose directional distributions were different but whose magnitude distributions lay in the range from 30 to 120 knots and resembled those shown in [**?**].

To create an upper baseline to compare with our planning algorithms, we created an "oracle" policy which could see true winds everywhere in the graph. Since no algorithm can even theoretically beat it, we report the results of all the other approaches as losses w.r.t. this baseline.

To isolate the influence of weather stations and wind patterns, we conduct two sets of experiments, each focusing on one of these factors. In the first set, we randomly vary the location of the weather stations while keeping the ground-truth wind pattern constant. The objective of these experiments is analyzing the sensitivity of different approaches to

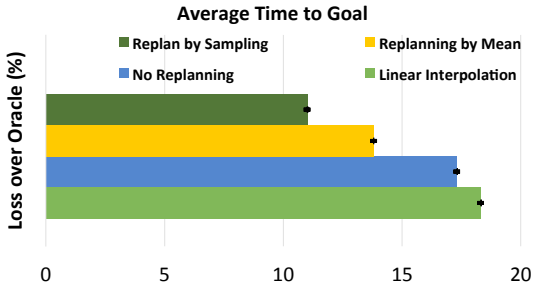

Figure 3: Comparison of the performances of different approaches in terms of the percentage loss (extra time taken) over the oracle policy.

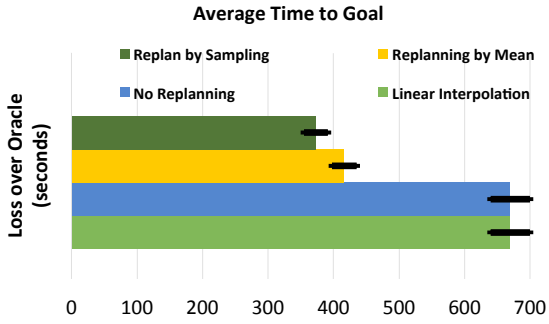

Figure 4: With 5 initial weather stations *Replan by Sampling* is better than *Replan by Mean* by 43 seconds on average while being 673.4 seconds better than *Linear Interpolation* and *No Replan* baselines.

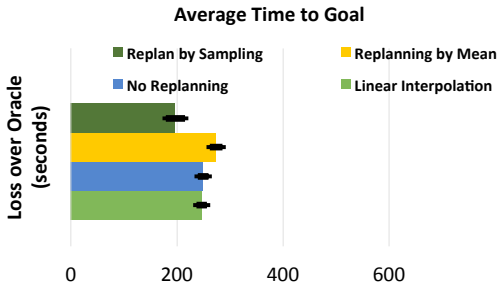

Figure 5: Performance comparison when 50 weather stations are available.

the apriori information provided by weather stations. The second set of experiments focuses on measuring the efficacy of replanning approaches against the baselines over a wide variety of wind patterns. It also lets us characterize the kind of conditions under which replanning has more gains over the baseline approaches.

Randomization over Wind Station Locations. In these experiments, we randomly vary the locations of 5 weather stations 400 times to explore their effect on planning approaches, while keeping the same ground-truth wind pattern. This pattern, shown in Figure **??**, is chosen to have sufficient variance to showcase the benefits of replanning and be fairly realistic. All approaches had access to the same wind information provided by the randomly located weather stations prior to planning. In addition, *Replan by Sampling* and *Replan by Mean* approaches updated their plans according to the additional true wind observations collected as the aircraft traversed the grid's edges.

Figure **??** shows the average flying times resulting from the plans computed by different approaches relative to the "oracle" planner's flying time as the percentage loss. In each trip, the oracle took 9845 seconds to get from the start to the goal in this particular graph. The average flying time for *Replan by Sampling* is 11% worse than the "Oracle" while the average time for "Replan by Mean" is 13.8% worse. The average times for *Linear-interpolation* and "No Replan" are 18.3% and 17.3% worse, respectively. On average, *Replan by Sampling*'s routes are 275 seconds shorter than *Replan by Mean*'s and 1801 seconds shorter than *Linear Interpolation*'s, potentially leading to significant fuel savings as well.

Randomization over Wind Patterns. In this set of experiments, we focus on investigating the performance of different approaches under a variety of wind patterns. We generated 500 random patterns, where each one represents different a combinations of wind directions and magnitudes. The start and goal locations were kept constant. We first ran all the approaches with 5 randomly selected initial weather stations held constant for all wind patterns. Figure **??** shows the average loss in time of each approach w.r.t. the "oracle" planner's average flight time. On average, *Replan by Sampling* outperforms *Replan By Mean*' by 43 seconds and beats *Linear Interpolation* and *No Replan* by 673.4 seconds.

In order to better understand where *Replan by Sampling* performs better than *Replan by Mean*, we binned the 500 runs by average speed of the true wind by every 5 m/s (9.7 knots). All the runs for which the average true wind speed fell into a particular bin were averaged and the mean loss over oracle was plotted in Figure **??**. The general trend is that both replanning based approaches perform increasingly better than *No-replan* and *Linear Interpolation* as the average true wind speed increases. In particular, at wind speeds of about 120 knots, some of the highest encountered by commercial aircraft at their cruising altitudes, *Replan by Sampling* saves $\sim$ 200 seconds compared to "Replan by Mean" and $\sim$ 800 seconds compared to the other two baselines.

Figure **??** shows the algorithms' performance as the number of initial wind stations is increased to 50. As expected, due to richer initial data GP can better model the wind, and the benefit from replanning decreases compared to the case of only 5 initial stations. Since there is less benefit in replanning due to more accurate prior information, the benefit of the approximate replanning approaches hinges on how well they can choose actions by taking the prior information and world uncertainty into account. Our results show that even in this case *Replan by Sampling* manages to perform better than *Replan by Mean* and the no-replanning approaches. They also show that *Replan by Mean* on average fails to provide any advantage compared to the no replanning approaches, since it cannot reason about uncertainty over wind patterns adequately.

7. CONCLUSION AND FUTURE WORK

We introduced the *Gaussian Traveler Problem* (GTP) for optimizing paths for domains with unknown edge costs that have dependencies governed by natural dynamics modeled by a Gaussian Process. We also presented a scalable approximate online algorithm for solving these problems. Casting the realistic domain of aircraft navigation as a GTP and comparing solutions for it yielded by the proposed algorithm and several baselines, showed that using GTPs in this set-

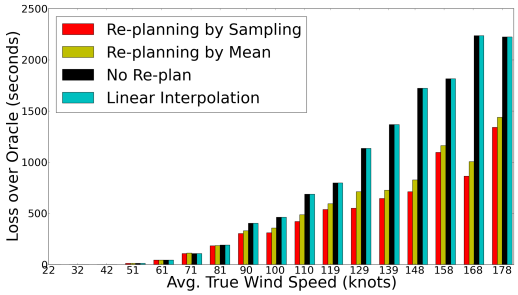

Figure 6: Performance loss over oracle as a function of average true wind speed. *Replan by Sampling* outperforms other methods for most of the wind speed values (lower bars denote better performance).

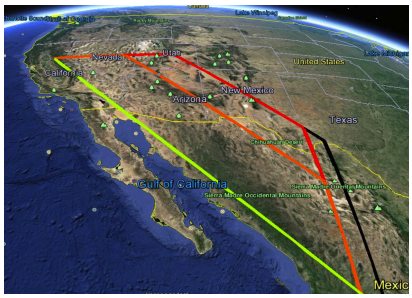

Figure 7: The path in black is the oracle plan from start (lower right corner) to goal (upper left corner), red is the *Replan by Sampling* plan which comes close to the oracle plan, orange is the *Replan by Mean* plan which in this case is worse than *Replan by Sampling* by 236 seconds. The green path is the plan taken by *Linear Interpolation* and *No Replan* which are both happen to be going straight for the goal in this example.

ting can lead to significant savings in fuel consumption and travel time.

We are currently exploring several directions for extending this work. We have focussed on planning approaches that use determinism for scalability, but there is opportunity for more sophisticated approaches such as Monte-Carlo planning to better represent the nature of sequential evidence collection while preserving algorithmic efficiency. We are working on generating more realistic simulations that are based on the trips of real-world aircraft. Moreover, our current model assumes that wind conditions do not change within the planning time horizon. In reality, during trips lasting several hours, predicted winds can change significantly. We plan to extend GTPs to handle *time-varying* natural dynamics by encoding temporal dependencies into the GP kernel, either explicitly with a time variable or implicitly by accounting for the agent's average travel time between different locations when learning spatial dependencies. We would also like to enable the agent's weather estimates to benefit from real-time observations made by other agents, e.g., aircraft overflying different parts of the continent. While incorporating these observations is as simple as updating the GP, the issue is that other agents may not be willing to share them voluntarily. For instance, weather readings are presently not shared between aircraft in a systematic way, and air traffic participants may charge for them. GTPs need an additional cost-benefit analysis mechanism to reason about this aspect. Finally, we are studying the challenges that arise from the deployment of these ideas for real-world flight planning. We believe that machine learning and planning techniques offer new opportunities for increasing the efficiency of existing practices in a wide array of practical domains, with the flight planning scenario presented in this paper being only one example.

8. REFERENCES

[1] *Private Pilot Manual*. Jeppesen Sanderson, 2001.

[2] A. G. Barto, S. Bradtke, and S. P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1):81–138, 1995.

[3] Z. Bnaya, A. Felner, and S. E. Shimony. Canadian traveler problem with remote sensing. In *IJCAI*, 2009.

[4] L. Boccia, P. Pace, G. Amendola, and G. Di Massa. Low multipath antennas for gnss-based attitude determination systems applied to high-altitude platforms. *GPS Solutions*, 2008.

[5] B. Bonet and H. Geffner. Solving POMDPs: RTDP-Bel vs. point-based algorithms. In *IJCAI*, 2009.

[6] G. Hollinger, A. Pereira, V. Ortenzi, and G. Sukhatme. Towards improved prediction of ocean processes using statistical machine learning. In *Robotics: Science and Systems Workshop on Robotics for Environmental Monitoring*, 2012.

[7] X. Jiang, B. Dong, L. Xie, and L. Sweeney. Adaptive gaussian process for short-term wind speed forecasting. In *ECAI*, 2010.

[8] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell. Gaussian Processes for object categorization. In *IJCV*, 2009.

[9] H. Kurniawati, D. Hsu, and W. Lee. Sarsop: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *RSS*, 2008.

[10] T. McGeer. Passive dynamic walking. *IJRR*, 9(2):62–82, 1990.

[11] A. Olsen. Pond-hindsight: Applying hindsight optimization to partially-observable markov decision processes. Master's thesis, Utah State University, 2011.

[12] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[13] C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84(1):127–150, 1991.

[14] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *IJCAI*, volume 3, pages 1025–1032, 2003.

[15] G. H. Polychronopoulos and J. N. Tsitsiklis. Stochastic shortest path problems with recourse. *Networks*, 27:133–143, 1996.

[16] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[17] D. Silver and J. Veness. Monte-carlo planning in large pomdps. In *NIPS*, pages 2164–2172, 2010.

[18] E. Sondik. *The Optimal Control of Partially Observable Markov Decision Processes*. PhD thesis, Stanford University, 1972.

[19] Y. Xie, K. Zhao, Y. Sun, and D. Chen. Gaussian processes for short-term traffic volume forecasting. *Journal of the Transportation Research Board*, 2165:69–78, 2010.