2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Adaptive Reliable Shortest Path in Gaussian Process Regulated Environments

Xuejie HOU[1], Hongliang GUO[1,†] and Yucheng ZHANG[2]

Abstract— **This paper studies the adaptive reliable shortest path (RSP) planning problem in a Gaussian process (GP) regulated environment. With the reasonable assumption that the travel times of the underlying transportation network follow a multi-variate Gaussian distribution, we propose two algorithms namely, Gaussian process reactive path planning (GPRPP), and Gaussian process proactive path planning (GP4), to generate online adaptive routing policies for the reliable shortest path. Both algorithms take advantage of the posterior analytical representation of GPs given past and/or imagined future observations of certain links in the network, and calculate the corresponding adaptive routing strategy for RSP. Theoretical analysis and simulation results (on Sioux Falls Network and Singapore road networks) show the superior performance of GPRPP and GP4 over that of the state of the arts.**

Index Terms— **adaptive path planning, Gaussian process, proactive path planning, GP4, GPRPP.**

I. INTRODUCTION

Stochastic shortest path (SSP) deals with the uncertain nature of traffic network's travel time, and offers a wide range of objectives for travellers to select, depending on his/her risk aversion attitude. Perhaps the most commonly used objective in SSP is to find a path with the least expected time (LET) [1], [2], [3], however, LET path focuses on minimizing the path's expected/mean travel time, which is only suitable for risk-neutral travellers. However, many travellers are risk averse, and prefer a more reliable path to the LET path, even they know that the LET path offers a smaller expected travel time [4].

This paper studies one of the SSP problems, namely, the reliable shortest path (RSP) planning problem, in GP regulated environments. To quantify RSP, many criteria have been proposed, including (1) maximize the stochastic on-time arrival (SOTA) probability [5], [6]; (2) minimize the α-reliable travel time [7], [8]; (3) minimize expected disutility (MED) [9], [10], [11]; and (4) minimize the mean-variance combination [12], or mean-standard deviation combination [13], [14], [15].

This paper assumes that the travel time over the underlying transportation network is spatially correlated, and we believe that this assumption is more realistic than the independent travel time assumption. For example, traffic congestions on a certain link will probably lead to high travel times over its upstream links. Moreover, we describe the joint distribution of travel time with a multi-variate Gaussian distribution, *i.e*. GP, which is the same assumption used in [12]. Empirical studies based on real-world traffic data show that the use of multi-variate normal distribution, *i.e*. GP, appears to reflect observed path travel time distributions very well [16].

GP is both flexible to capture the spatial correlations of travel time over the underlying transportation network, and convenient to derive posterior distributions when given a subset of samples. We take advantage of the analytical representation of GP posterior, and propose a GP-based reactive path planning algorithm which updates the posterior travel time distribution whenever the ego vehicle collects a subset of samples from the environment, and delivers to the ego vehicle an updated RSP based on the posterior travel time distribution. We further propose a proactive path planning method, which plans the RSP even before the ego vehicle enters the link. A sampling method is proposed to estimate the RSP metric even before the ego vehicle enters a certain link, and the proactive path planner will decide the best link that the vehicle should take. Computational complexity is analyzed theoretically, and the performance enhancement over state of the arts is validated experimentally.

The contributions of the paper can be summarized as follows: (1) we propose two adaptive RSP planning algorithms within GP-regulated environment, namely GPRPP and GP4, both of which achieve better performance than that of state of the arts; (2) deterministic sampling method is utilized to ensure the applicability of GP4 even before the ego vehicle collects the travel time sub-samples; and (3) simulation results on a canonical transportation testbed (Sioux Falls Network) and a realistic road network (Singapore arterial network) show the superior performance of GPRPP and GP4.

II. LITERATURE REVIEW

This section provides a brief review over the aforementioned RSP objectives, namely (1) MED, (2) SOTA, (3) α-reliable shortest path and (4) mean-variation/mean-std minimization, and the corresponding solution algorithms.

The MED RSP introduces a disutility function representing the weight of a path as a function of the arrival time, and seeks for a path with MED. The exponential disutility function is proposed and solved in [17] for stationary and independent travel time distributions. In [9], the authors extend the MED RSP algorithm to application scenarios with time-dependent and correlated travel time. The quadratic disutility function which requires relaxation-based pruning for the exact solution is reported in [18], and later, the authors

[1]University of Electronic Science and Technology of China, China 201821060832@std.uestc.edu.cn, guohl@uestc.edu.cn

[2]Intelligent Agricultural Machinery Equipment Laboratory of Chinese Academy of Sciences, Beijing, China zhangyucheng@ict.ac.cn

[†]Hongliang GUO is the corresponding author.

978-1-7281-6211-9/20/$31.00 ©2020 IEEE

solved the RSP MED problem with piecewise linear convex disutility function [11].

The α-reliable RSP, as defined in [7], aims at finding a path with the minimal high percentile travel time. Chen and Ji propose to use genetic algorithms to find such a path [7]. The spatially dependent use case for α-reliable RSP is solved in [19].

The SOTA criterion is first proposed by Frank, which aims at finding a path with the maximal probability of on time arrival [5]. The corresponding solution algorithms can be roughly categorized into three groups, namely iterative dynamic programming [20], integer programming [21], and Lagrangian relaxation [22]. Fan *et al*., proposed a dynamic programming technique, which iteratively optimize the SOTA objective, and delivers the optimal next visiting node as the routing policy. The method is applicable only to discrete travel time distributions and the output of the SOTA path is a dynamic routing policy instead of a prior path [20]. Cao *et al*., propose a data-driven SOTA path calculation method, which transforms the problem into an integer linear programming approach. The method is able to solve the SOTA problem with various distribution types of travel time. Lagrangian relaxation is a popular polynomial algorithm solver for SOTA [21]. For example, Cao *et al*.use partial Lagrangian method to relax the original SOTA problem into a series of LET path finding problem, and the resulting approximate solution has a polynomial computational complexity [22].

The mean-variance or mean-std RSP incorporates the path's travel time variance or standard deviation directly into the formulation, either as a constraint [23] or as an additional term in the objective function [12], [24]. Sen *et al*. propose a polynomial time algorithm to solve the mean-variance RSP problem, in which the optimal path is guaranteed within a pre-calculated set of paths [12]. The algorithms for mean-std RSP can be roughly categorized into three groups: (1) formulate the problem into a mixed integer non-linear programming (MINLP) problem and use off-the-shelf optimization solvers for the solution [25], [10][1]; (2) transform the problem into a series of easily solvable problems for iterative solutions [14], [26]; (3) apply Lagrangian relaxation and duality theory for approximate solution with duality gap analysis [27], [28], [15].

The difference between our algorithm and canonical mean-std/mean-variance RSP algorithms is that we output a routing policy which only delivers the next node/link to visit and relay on the complete path calculation to later decision stages, while most state of the arts are delivering a complete a priori path. We will show in the Methodology Section that postponing the complete path calculation to later stages enables a better policy as later stage information helps with the posterior GP update.

[1]Note that some MINLP approaches also make use of (partial) Lagrangian relaxation for approximate solutions.

III. Problem Formulation

This section introduces the notations used throughout the paper, followed by the mean-variance and mean-std RSP problem statement. Then, we lay down two reasonably assumptions used in the paper, which 'naturally' avoid the cycle-containing RSP solution without explicitly stating the cycle-elimination constraints.

A. Notations

Let $\mathcal{G}(\mathcal{N},\ \mathcal{A})$ represent a directed and connected transportation network, where $\mathcal{N}$ ($|\mathcal{N}| = n$) is the set of nodes and $\mathcal{A}$ ($|\mathcal{A}| = m$) is the set of links. Let $\mathbf{c} \in \mathcal{R}^m$ be a random variable (RV) vector, which represents the joint travel time distribution over $\mathcal{G}$. This paper assumes that $\mathbf{c} \sim \mathcal{N}(\mu, \mathbf{\Sigma})$, where $\mu \in \mathcal{R}^m$ is the mean of $\mathbf{c}$, and $\mathbf{\Sigma} \in \mathcal{S}_{++}^{m\times m}$ is the positive co-variance matrix capturing the correlation between link travel times. It is worth noting that we are looking at the stationary but stochastic environment in which the travel time distribution does not change over time.

Other related variable notations are listed as follows: (1) $i,\ j \in \mathcal{N}$ refers to the node index with r representing the origin and s representing the destination; (2) $ij,\ kl \in \mathcal{A}$ refers to link index; (3) $x_{ij} \in \{0, 1\}$ is a binary decision variable, when $x_{ij} = 1$, it means that link ij is selected for the RSP; (4) σ_{ij} is the standard deviation of link ij's travel time, $\text{Cov}(ij, kl)$ is the co-variance of travel times between link ij and link kl (both σ_{ij}^2 and $\text{Cov}(ij, kl)$ are elements of the co-variance matrix $\mathbf{\Sigma}$); and (5) $\zeta \geq 0$ is the reliability coefficient representing the user's risk aversion attitude.

B. Problem Statement

The output of our algorithm is a routing policy which minimizes the mean-std or mean-variance linear combination of the reliable shortest path. The optimal routing policy depends on the backbone RSP problem formulation which outputs the a priori optimal path, thus we present the backbone problem formulations for mean-variance RSP and mean-std RSP in (P1) and (P2), respectively:

(P1):

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta \boldsymbol{x}^\top \Sigma \boldsymbol{x} \\ \text{s.t.} \quad & \sum_{j:ij\in\mathcal{A}} x_{ij} - \sum_{k:ki\in\mathcal{A}} x_{ki} = \begin{cases} 1, & i = r \\ -1, & i = s \\ 0, & i \in \mathcal{N} - \{r, s\} \end{cases}, \\ & x_{ij} \in \{0, 1\}. \end{aligned}$$

(P2):

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{\mu}^\top \boldsymbol{x} + \zeta \sqrt{\boldsymbol{x}^\top \Sigma \boldsymbol{x}} \\ \text{s.t.} \quad & \sum_{j:ij\in\mathcal{A}} x_{ij} - \sum_{k:ki\in\mathcal{A}} x_{ki} = \begin{cases} 1, & i = r \\ -1, & i = s \\ 0, & i \in \mathcal{N} - \{r, s\} \end{cases}, \\ & x_{ij} \in \{0, 1\}. \end{aligned}$$

Note that in both (P1) and (P2), the constraints are flow balancing constraints with boolean decision variables. There have been many algorithms for efficient solutions to (P1) and (P2), and one can refer to [28] for the mean-std RSP review and [24] for the mean-variance RSP review. In this paper, we will use (P1) and (P2) as the backbone problem formulation and rely on one of the canonical solvers, *i.e*. MINLP for the solution. We take advantage of the GP assumption of the underlying transportation network, and transform the backbone problem (P1) and (P2) to reach the optimal routing policy.

C. Assumptions

In this subsection, we lay down several reasonable assumptions so as to avoid the 'cycling' problem, *i.e*. the optimal path contains cycles.

Assumption 1 (Travel-Time Assumption)**.** *The mean travel times of all the links are strictly positive, i.e.* $\boldsymbol{\mu} \succ \mathbf{0}$.

Assumption 2 (Cycle Co-variance Assumption)**.** *Removing a cycle in a path results in a path whose total variance is strictly less than the original path.*

It is intuitive to justify that Assumption 1 is reasonable, as any link that the traveller travels on would consume a time, therefore, the mean-travel time for any link is greater than zero. Additionally, Assumption 2 is also reasonable, because travel on additional links can only add uncertainties, and hence increase the variance.

Theorem 1. *The optimal solution to (P1) or (P2) cannot be a cycle-containing path.*

Without loss of generality, we only sketch the proving process of the case with one cycle in the path. The proof sketch process is as follows: we start with the assumption that the optimal path to (P1) or (P2) is a one-cycle containing path. We remove the cycle contained in the path, and the resulting cycle-free path is still a valid path connecting the origin (r) with the destination (s). Then, we verify that the cycle-free path has a strictly lower mean travel time and variance, which indicates that the cycle-free path holds a smaller objective value than the corresponding one-cycle containing path. Then, the starting assumption is invalid, and hence the proof is completed.

Next, we give out the following two theorems without proof due to page limitations and the proving process is straightforward after referring to [29].

Theorem 2. *In GP-regulated transportation network, the 'risk-averse' α-reliable RSP ($\alpha \geq 0.5$) is equivalent to a mean-std RSP, with $\zeta = \Psi^{-1}(\alpha)$, where $\Psi(x)$ is the cumulative distribution function (CDF) of the standard normal distribution.*

Theorem 3. *In GP-regulated transportation network, the 'risk averse' SOTA RSP*[2] *with deadline T is equivalent to a mean-std RSP, with a specific ζ, which returns the optimal path with the objective value equal to T, and the bipartite algorithm can be used to identify ζ.*

[2]The 'risk averse' SOTA RSP refers to the SOTA problem with a deadline T larger than the LET path's expected travel time.

IV. Methodology

This section introduces the GPRPP and GP4 algorithm for mean-variance RSP (P1) and mean-std RSP (P2) solutions. Both GPRPP and GP4 will output an adaptive routing policy instead of a priori path. Since there exist many efficient solves for mean-variance RSP [12], [24] and mean-std RSP [28], [10], [15], [26], we will rely on those state of the arts as the backbone solution for (P1) and (P2), and take advantage of the GP's prior-posterior analytical relationship to output a better routing policy.

A. Parameter Update for Posterior Gaussian Process

In both GPRPP and GP4, we take advantage of the analytical posterior Gaussian process expressions for a better routing policy. Therefore, this subsection introduces the posterior GP representation form given a subset of samples.

Assume $X \in \mathcal{R}^n$ is a GP, *i.e.* $X \sim \mathrm{GP}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and X is partitioned into two subsets: $X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}$, with $X_1 \in \mathcal{R}^p$ and $X_2 \in \mathcal{R}^{(n-p)}$. Correspondingly, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are partitioned as $\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix}$, respectively. Then the distribution of X_1 conditional on $X_2 = \boldsymbol{x_2}$ is multivariate normal $X_{1|X_2=x_2} \sim \mathcal{N}(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2})$ where

$$\boldsymbol{\mu}_{1|2} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{x_2} - \boldsymbol{\mu}_2), \tag{1}$$

and the posterior co-variance matrix

$$\boldsymbol{\Sigma}_{1|2} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}. \tag{2}$$

B. Gaussian Process Reactive Path Planning (GPRPP)

The intuition of GPRPP is straightforward: it will take the backbone solution to (P1) or (P2) as the current routing policy, and suggest the first link based on the ego vehicle's current location. When the ego vehicle takes the suggested action, *i.e.* travels on the suggested link (x_{ij}), a realization of that link c_{ij} (denoted as r_{ij}) will be observed, and the posterior distribution of all the other links in the underlying transportation network can be updated as: $c_{1|c_{ij}=r_{ij}} \sim \mathcal{N}(\boldsymbol{\mu}_{1|2}, \boldsymbol{\Sigma}_{1|2})$, where $\boldsymbol{\mu}_{1|2}$ and $\boldsymbol{\Sigma}_{1|2}$ can be easily computed after referring to Eq. 1 and Eq. 2, respectively.

The flow process of GPRPP is depicted in **Algorithm 1**. When supplied with required parameters, *i.e.* $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and ζ, GPRPP will loop between calling the backbone solver, and the following three steps: (1) execute the link; (2) collect the travel-time realization over the link; (3) calculate posterior travel-time distribution based on the collected sample.

C. Gaussian Process Proactive Path Planning (GP4)

The previous subsection introduces GPRPP, which uses the collected travel time for posterior travel time distribution calculation, and recursively calls the backbone solver for an adapted routing policy. In this subsection, we introduce a proactive routing policy, which determines the best routing policy even before the vehicle enters the link and collects

Algorithm 1: The GPRPP Algorithm Flow Process

Input: GP parameters for the underlying transportation network ($\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$), reliability coefficient ζ, current node r, destination node s.

Output: GPRPP adaptive routing policy.

Set counter $k = 0$, $\boldsymbol{\mu}_k = \boldsymbol{\mu}$, $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$, $r_k = r$.
while $r_k \neq s$ **do**
    call the backbone solver for (P1) or (P2) with parameters $\boldsymbol{\mu}_k$,$\boldsymbol{\Sigma}_k$,ζ, r_k and s;
    get the RSP path $\boldsymbol{x}_k$;
    Execute the RSP selected link ($x_{r_k q}$) which starts at node r_k and ends at node q;
    collect the sampled travel time $c_{r_k q}$;
    calculate the posterior distribution parameters $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ according to Eq. 1 and Eq. 2;
    $k \leftarrow k + 1$;
    $\boldsymbol{\mu}_k \leftarrow \boldsymbol{\mu}_p$;
    $\boldsymbol{\Sigma}_k \leftarrow \boldsymbol{\Sigma}_p$;
    $r_k = q$;
Final.

the travel-time data. Note that we use (P1) as the routing objective, and we can simply change the backbone solver to (P2) if (P2) serves as the objective.

We denote the deterministic adaptive routing policy as π, which will output a determined next visiting node ($\pi(r)$) based on the current node (r), destination node (s), and GP parameters ($\boldsymbol{\mu}$, and $\boldsymbol{\Sigma}$). The policy is adaptive with respect to the updated GP parameters. We define the value of a node as the expectation of the mean-std value if we let the vehicle follow the proactive routing policy. Thus, the recursive relationship between the value for the current node value and the value of the next node is expressed as follows:

$$V^{\pi}(r; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathbb{E}\left(c_{r,\pi(r)} + V^{\pi}(\pi(r); \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\right), \tag{3}$$

where $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ are computed according to Eq. 1 and Eq. 2, based on the instantiated value of $c_{r,\pi(r)}$. The value for the best policy out of the whole policy space can be recursively represented as:

$$V^{*}(r; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \min_{\pi} \mathbb{E}\left(c_{r,\pi(r)} + V^{*}(\pi^{*}(r); \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\right). \tag{4}$$

Eq. 4 is the Bellman equation for GP4, and theoretically, we can use policy iteration or value iteration to find the optimal policy. However, recursively bootstrapping the value function is computationally complex due to the large state space. In this paper, we use the backbone solver as mentioned in the last subsection to estimate the value of a node, *i.e.* the second part of the right hand side (RHS) of Eq. 4. In this case, we can calculate V^* through evaluating the RHS of Eq. 4 and hence decide the optimal policy.

Now, we fix a deterministic policy π, and calculate the term $\mathbb{E}\left(c_{r,\pi(r)} + V^{*}(\pi(r); \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\right)$. Note that $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ are calculated through Eq. 1 and Eq. 2, and both terms depend on the sampled travel time $c_{r,\pi(r)}$. The backbone solver ensures that when supplied with related parameters, *i.e.* $\boldsymbol{\mu}_p$, $\boldsymbol{\Sigma}_p$, r, s, and ζ, the estimation of V^*, *i.e.*, $\hat{V}^*$ can be calculated. Since there is no analytical representation form for explicit routing policy evaluation, we approximate the value of the routing policy with a computer-implementable term. The derivation process is as follows:

$$\begin{aligned}
&\mathbb{E}\left(c_{r,\pi(r)} + V^{*}(\pi(r); \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\right) \\
&= \int_{-\infty}^{\infty} \big(x + V^{*}(\pi(r); \boldsymbol{\mu}_p(x), \boldsymbol{\Sigma}_p(x))\big) p(x)\, \mathrm{d}x \\
&= \int_{-\infty}^{\infty} \big(x\, p(x) + V^{*}(\pi(r); \boldsymbol{\mu}_p(x), \boldsymbol{\Sigma}_p(x))\, p(x)\big)\, \mathrm{d}x \\
&= \mathbb{E}(X) + \int_{-\infty}^{\infty} V^{*}(\pi(r); \boldsymbol{\mu}_p(x), \boldsymbol{\Sigma}_p(x))\, p(x)\, \mathrm{d}x \\
&= \mathbb{E}(c_{r,\pi(r)}) + \int_{-\infty}^{\infty} V^{*}(\pi(r); \boldsymbol{\mu}_p(x), \boldsymbol{\Sigma}_p(x))\, p(x)\, \mathrm{d}x \\
&\approx \mathbb{E}(c_{r,\pi(r)}) + \int_{-\infty}^{\infty} \hat{V}^{*}(\pi(r); \boldsymbol{\mu}_p(x), \boldsymbol{\Sigma}_p(x))\, p(x)\, \mathrm{d}x \\
&\approx \mathbb{E}(c_{r,\pi(r)}) + \sum_{i=1}^{N} \hat{V}^{*}\big(\pi(r); \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p\big)\, p(x_i)\, \delta x. \qquad (5)
\end{aligned}$$

The essence of the transformation is to 'deterministically' pre-sample (which we call deterministic sampling[3]) the travel time data over the 'to-be-executed' link, and approximate the integral with a summation term. We give the following theorem, whose proof is straightforward and omitted here.

Theorem 4. *Suppose that the backbone solver has an accurate node value estimation, i.e.* $\hat{V}^* = V^*$*, and as* $N \to \infty$*, i.e.,* $\delta x \to 0$*, Eq. 5 approaches* $V^{\pi}(r; \boldsymbol{\mu}, \boldsymbol{\Sigma})$*, which means:*

$$\lim_{N \to \infty} \sum_{i=1}^{N} \hat{V}^{*}\big(\pi(r)\big)\, p(x_i)\, \delta x + \mathbb{E}(c_{r,\pi(r)}) = V^{\pi}(r; \boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

The essence of GP4 is to relay the value estimation of a given node to its succeeding node. The vehicle will go through N 'proactively' sampled travel time along the chosen link, and evaluate the value of the succeeding node based on the posterior Gaussian distribution with the backbone solver. Then, GP4 will calculate the expectation of the mean-std (or mean-variance) of the policy. Here, the word 'proactive' means that the sampled travel time must be within the corresponding interval δx. The algorithm flow process of GP4 is depicted in **Algorithm 2**.

V. Computational Complexity Analysis

This section analyzes the computational complexity of GPRPP and GP4. The computational cost of an operation can often be expressed through the number of floating-point operations (flops). A flop is defined as an addition, subtraction, multiplication or division of two floating-point numbers. To evaluate the complexity of an algorithm, we

[3]We partition the travel time sampling space into N blocks, and 'deterministically' pick the center of each block as the sampled travel time. Compared to the normal sampling method, deterministic sampling avoids the sampling variance.

Algorithm 2: The GP4 Algorithm Flow Process

Input: GP parameters for the underlying transportation network ($\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$), reliability coefficient ζ, current node r, destination node s, partition number N.

Output: GP4 adaptive routing policy.

Set counter $k = 0$, $\boldsymbol{\mu}_k = \boldsymbol{\mu}$, $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$, $r_k = r$.
while $r_k \neq s$ **do**
  $\forall j \in \mathcal{N}$ such that $r_k j \in \mathcal{A}$: calculate the estimated node value according to Eq. 5, if the deterministic policy π selects j as the next node to visit;
  During the calculation process, call the backbone solver for (P1) N times with estimated posterior GP parameters ($\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$);
  Select the node j which minimizes$\big(\mathbb{E}(c_{r,j}) + \sum_{i=1}^{N} \hat{V}^*(j; \boldsymbol{\mu}_p(x_i), \boldsymbol{\Sigma}_p(x_i))\, p(x_i)\, \delta x\big)$;
  Execute the GP4 selected link ($x_{r_k j}$) which starts at node r_k and ends at node j;
  collect the sampled travel time $c_{r_k j}$;
  calculate the posterior distribution parameters $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ according to Eq. 1 and Eq. 2;
  $k \leftarrow k + 1$;
  $\boldsymbol{\mu}_k \leftarrow \boldsymbol{\mu}_p$;
  $\boldsymbol{\Sigma}_k \leftarrow \boldsymbol{\Sigma}_p$;
  $r_k = j$;
Final.

count the total number of flops; express it as a function (usually a polynomial) of the dimensions of the matrices and vectors involved, and simplify the expression by ignoring all terms except the leading terms.

In both GPRPP and GP4, there are two basic operations, namely (1) posterior GP parameter computation, and (2) backbone solver for (P1) or (P2). The equations for posterior GP update are Eq. 1 and Eq. 2, and the computational complexity is $\mathcal{O}(|\mathcal{A}|^p)$, where $p = 4$ for posterior mean computation and $p = 5$ for posterior co-variance computation. It is worth noting that the computational complexity only depends on the number of arcs ($|\mathcal{A}|$), but does not depend on the number of nodes ($|\mathcal{N}|$) in the network. The backbone solver for (P1) and (P2) depends on the specific research works, and for (P1), there are polynomial solvers for the exact solution, *e.g.* [12], and for (P2), polynomial solvers only exist for approximate solutions, and approximate solutions with polynomial computational complexity exist, *e.g.* [28], [10]. This paper uses $\mathcal{O}(|\mathcal{N}|^{p_1}|\mathcal{A}|^{p_2})$ as a general form for the backbone solver's computational complexity representation.

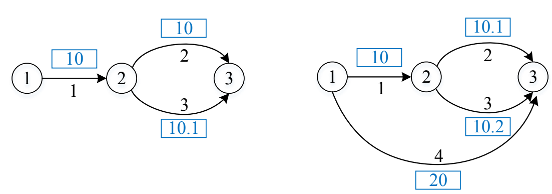
Examining Algorithm 1, we can see that GPRPP essentially loops between the backbone solver and posterior GP computation, and the maximal number of loops is $|\mathcal{N}|$, thus the computational complexity of GPRPP is $|\mathcal{N}| \times \mathcal{O}(|\mathcal{N}|^{p_1}|\mathcal{A}|^{p_2+p}) = \mathcal{O}(|\mathcal{N}|^{p_1+1}|\mathcal{A}|^{p_2+p})$. The computation process of GP4 is similar to that of GPRPP, but inside the node value evaluation procedure, it will proactively sample the travel time data, and calculate the expected mean-std value, thus the computational complexity is represented as $\mathcal{O}(N|\mathcal{N}|^{p_1+1}|\mathcal{A}|^{p_2+p})$.

VI. Two Illustrative Examples

In this section, we present two simple yet illustrative examples to show how and why GPRPP and GP4 outperform state of the arts. In both examples, we set $\zeta = 0$, and in this case, both (P1) and (P2) are having the same objective, which is the least expected time (LET). Fig. 1(a) shows a transportation network with 3 nodes and 3 links, and the vehicle is supposed to navigate from node 1 to node 3. The corresponding mean travel time of each link is marked in the figure, and the covariance matrix ($\boldsymbol{\Sigma}$) is set as:

$$\boldsymbol{\Sigma} = \begin{bmatrix} 2 & -1 & 1 \\ -1 & 2 & 0 \\ 1 & 0 & 1 \end{bmatrix}. \qquad (6)$$

Since $\zeta = 0$, both (P1) and (P2) become an LET path seeking problem. We use Dijkstra's algorithm [30] as the backbone solver, and the optimal a prior path is $\boldsymbol{x} = (1, 0, 1)^\top$, which selects link 1 and link 2 to connect node 1 and node 3. However, if we run GPRPP; collect the sampled travel time data over link 1, and calculate the posterior travel time distribution of link 2 and link 3, we would find that there is roughly a 50% chance that link 3 becomes the better route choice. For example, if link 1's sampled travel time is $c_1 = 8$, we calculate the posterior mean of link 2 ($\mu_{2|1}$) and link 3 ($\mu_{3|1}$) according to Eq. 1, and the result is $\mu_{2|1} = 11$, $\mu_{3|1} = 9.1$. In this case, it is apparent that the ego vehicle should select link 3 instead of link 2 to reach node 3. In fact, after simple derivation, we can find that as long as the sampled travel time of link 1 is less than 9.9, link 3 would be a better route choice than link 2. Considering that link 1's mean travel time is 10, it means that roughly 50% of the time, link 3 becomes the optimal route choice.



(a) Example for GPRPP (b) Example for GP4

Fig. 1: Two simple yet illustrative transportation networks: the vehicle is supposed to go from node 1 to node 3, and the boxed number over each link refers to the link's mean travel time.

Fig. 1(b) shows a slightly more complex transportation network with 3 nodes and 4 links, whose mean travel times are marked in the figure. The vehicle is also supposed to navigate from node 1 to node 3. The covariance matrix ($\boldsymbol{\Sigma}$)

is set as:

$$\mathbf{\Sigma} = \begin{bmatrix} 2 & -1 & 1 & 0 \\ -1 & 2 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

With either Dijkstra's algorithm or GPRPP, link 4 will be selected to navigate the vehicle from node 1 to node 3. However, GP4 will proactively sample the travel time of link 1 and link 4, and evaluate the resulting expected travel time. After evaluating Eq. 5 with $N = 100$, GP4 selects link 1 (instead of link 4) to execute with expected total travel time at 19.9, while link 4's expected travel time is 20. This simple example shows that even for the LET path seeking problem, GP4 is able to yield better routing strategies than both GPRPP and other state of the arts.

VII. Simulation Results and Analysis

In this section, we first test the proposed two algorithms (GPRPP and GP4) in the Sioux Falls Network (Fig. 2), which is a canonical testbed for transportation studies. Then, we present the experimental results on a realistic network (Singapore arterial road network). For (P1)'s backbone solver, we use the algorithm proposed in [12], which has a polynomial computational complexity, and for (P2)'s backbone solver, we use the most recently proposed mean-std RSP solver [28].

A. Sioux Falls Network

Sioux Falls Network (Transportation Test Problems) has 24 nodes and 76 links with associated link travel times. The links' mean travel times are set the same as Table 1 in [14], and the covariance matrix is randomly generated with the square root of each diagonal element within 0.15 of the expected travel time of the corresponding link[4], and each off-diagonal element between -0.225 and $+0.0225$ of the product of the corresponding two links' expected travel time. Both algorithms are tested to generate RSP from node 1 to 15. We use the algorithm proposed in [28] as the backbone solver, and N is set to be 100 for the GP4 algorithm. As the routing algorithm depends on the sampled travel time along the executed links, we run the algorithm for 1000 independent times and the average performance metric is reported in Fig. 3.

We test the performance of GPRPP and GP4 against the backbone solver for both the mean-var objective (Fig. 3(a)) and the mean-std objective (Fig. 3(b)). For the backbone solver, we use the algorithm proposed in [12] for the mean-var objective, and use the algorithm in [28] for the mean-std objective. In the figure, we can see that the performance metric increases for all of the algorithms as we increase the reliability coefficient (ζ), but both GP4 and GPRPP are having lower values than that of the backbone solver.

[4]This is to align with the experimental settings in [14], which generates the std values within 0.15 of the expected link travel time.

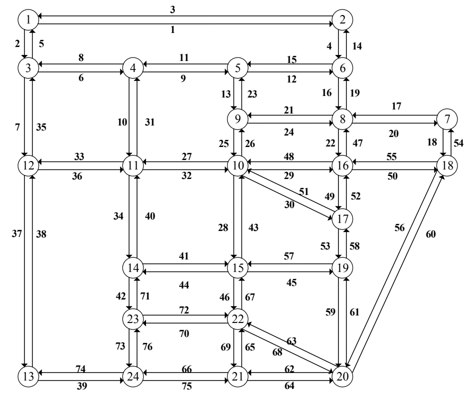
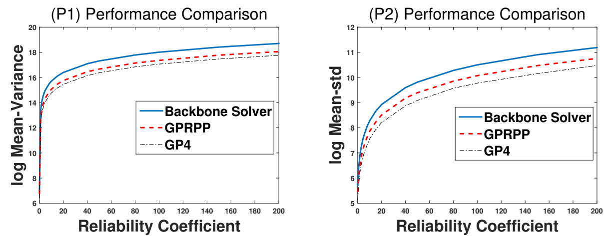


Fig. 2: Sioux Falls Network

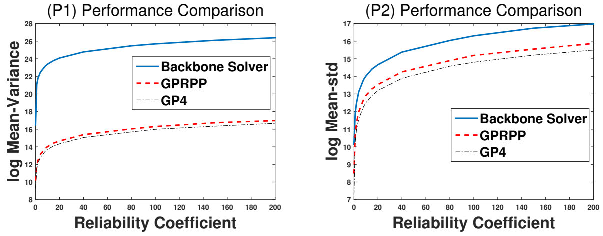


(a) Mean-var objective (P1) (b) Mean-std objective (P2)

Fig. 3: Performance comparison of GPRPP and GP4 against the backbone solver for the mean-var objective (P1) and mean-std objective (P2). We report the '$\ln$' value of the performance metric for better visualization purpose.

B. Singapore Arterial Road Network

Both GPRPP and GP4 are also tested on a realistic use case: Singapore arterial road network. To construct the transportation network, we extract all the nodes and links of the whole Singapore road network from Open Street Map (OSM) [31], and then remove some small road segments while keep the major/arterial roads (including highways). The Singapore arterial road network consists of $6,476$ nodes and $10,253$ links. Although we do not have the mean travel time statistics for the Singapore network, OSM offers the length of each road segment and the corresponding speed limit. In this paper, we assume that the vehicles normally travel at half of the speed limit, and the mean travel time can be calculated as $\mu_{ij} = 2l_{ij}/v_{ij}$, where l_{ij} is the length of link ij, and v_{ij} is the corresponding speed limit. The covariance matrix is generated randomly following the same mechanism as described in the Sioux Falls network subsection. We randomly generate 20 OD pairs, and for each OD pair, we run the GPRPP and GP4 algorithm for 200 independent times and report the corresponding performance in Fig. 4. From the figure, we can see that both GPRPP and GP4 outperform the state of the arts (backbone solver in the figure), and GP4, in general, is better than GPRPP. The computation time for each decision making step heavily depends on how the backbone solver behaves, thus we do not report the actual values here, but they are at the same time scale of the backbone solver as analyzed in the computational complexity analysis section.



(a) Mean-var objective (P1) (b) Mean-std objective (P2)

Fig. 4: Performance comparison of GPRPP and GP4 against the backbone solver for the mean-var objective (P1) and mean-std objective (P2) on Singapore arterial network. 'ln' values are used for better visualization purpose.

VIII. Conclusion and Future Works

This paper presents GPRPP and GP4 for reliable shortest path planning. By reasonably assuming that the travel time distribution over the underlying transportation network follows a GP, both algorithms are able to output an adaptive routing strategy which is more reliable than state-of-the-art solutions. Illustrative use cases show why and how GPRPP and GP4 outperform state of the arts. Simulation results on canonical testbed as well as a realistic road network verify the superior performance of both algorithms.

In the future, we plan to replace the GP assumption with the log-GP distribution. Real travel time data is strictly positive, but GP can never eliminate the negative travel time samples. In the meanwhile, we will propose a robust GP path planning algorithm in which the GP parameters are within a certain range instead of precisely known in the current paper. More experiments exploring what covariance function benefits the GPRPP/GP4 algorithm will also be conducted.

Acknowledgement

This research was supported by the Zhejiang Lab's International Talent Fund for Young Professionals.

References

[1] L. Fu, "An adaptive routing algorithm for in-vehicle route guidance systems with real-time information," *Transportation Research Part B: Methodological*, vol. 35, no. 8, pp. 749 – 765, 2001.

[2] E. D. Miller-Hooks and H. S. Mahmassani, "Least expected time paths in stochastic, time-varying transportation networks," *Transportation Science*, vol. 34, no. 2, pp. 198–215, 2000.

[3] R. Hall, "The fastest path through a network with random time-dependent travel times," *Transportation Science*, vol. 20, no. 3, pp. 182–188, 1986.

[4] D. Brownstone, A. Ghosh, T. Golob, C. Kazimi, and D. van Amelsfort, "Drivers' willingness-to-pay to reduce travel time: Evidence from the san diego i-15 congestion pricing project," *Transportation Research Part A: Policy and Practice*, vol. 37, pp. 373–387, 05 2003.

[5] H. Frank, "Shortest paths in probabilistic graphs," *Operations Research*, vol. 17, pp. 583–599, 1969.

[6] Y. Nie and Y. Fan, "Arriving-on-time problem: Discrete algorithm that ensures convergence," *Transportation Research Record*, vol. 1964, no. 1, pp. 193–200, 2006.

[7] A. Chen and Z. Ji, "Path finding under uncertainty," *Journal of Advanced Transportation*, vol. 39, pp. 19 – 37, 2005.

[8] P. Chen, R. Tong, G. Lu, and Y. Wang, "The α-reliable path problem in stochastic road networks with link correlations: A moment-matching-based path finding algorithm," *Expert Systems with Applications*, vol. 110, pp. 20 – 32, 2018.

[9] H. Huang and S. Gao, "Optimal paths in dynamic networks with dependent random link travel times," *Transportation Research Part B: Methodological*, vol. 46, no. 5, pp. 579 – 598, 2012.

[10] M. Shahabi, A. Unnikrishnan, and S. Boyles, "Robust optimization strategy for the shortest path problem under uncertain link travel cost distribution," *Computer-Aided Civil and Infrastructure Engineering*, vol. 30, no. 6, pp. 433–448, 2015.

[11] I. Murthy and S. Sarkar, "Stochastic shortest path problems with piecewise-linear concave utility functions," *Management Science*, vol. 44, no. 11, pp. 125–136, 1998.

[12] S. Sen, R. Pillai, S. Joshi, and A. K. Rathi, "A mean-variance model for route guidance in advanced traveler information systems," *Transportation Science*, vol. 35, no. 1, pp. 37–49, 2001.

[13] T. Uchida and Y. Iida, "Risk assignment: A new traffic assignment model considering the risk of travel time variation," in *12th International Symposium on Transportation and Traffic Theory*, 1993, pp. 89–105.

[14] A. Khani and S. D. Boyles, "An exact algorithm for the mean-standard deviation shortest path problem," *Transportation Research Part B: Methodological*, vol. 81, pp. 252 – 266, 2015.

[15] Y. Zhang, Z.-J. M. Shen, and S. Song, "Lagrangian relaxation for the reliable shortest path problem with correlated link travel times," *Transportation Research Part B: Methodological*, vol. 104, pp. 501 – 521, 2017.

[16] H. Rakha, I. El-Shawarby, M. Arafeh, and F. Dion, "Estimating path travel-time reliability," in *IEEE Intelligent Transportation Systems Conference*, 2006, pp. 236 – 241.

[17] A. Eiger, P. Mirchandani, and H. Soroush, "Path preferences and optimal paths in probabilistic networks," *Transportation Science*, vol. 19, no. 1, pp. 75–84, 1985.

[18] I. Murthy and S. Sarkar, "A relaxation-based pruning technique for a class of stochastic shortest path problems," *Transportation Science*, vol. 30, no. 3, pp. 220–236, 1996.

[19] B. Y. Chen, W. Lam, and Q. Li, "Efficient solution algorithm for finding spatially-dependent reliable shortest path in road networks," *Journal of advanced transportation*, vol. 50, pp. 1413–1431, 11 2016.

[20] Y. Y. Fan, R. E. Kalaba, and J. E. Moore, "Arriving on time," *Journal of Optimization Theory and Applications*, vol. 127, no. 3, pp. 497–513, 2005.

[21] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, "Finding the shortest path in stochastic vehicle routing: A cardinality minimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1688–1702, 2015.

[22] Z. Cao, H. Guo, J. Zhang, D. Niyato, and U. Fastenrath, "Improving the efficiency of stochastic vehicle routing: A partial Lagrangian multiplier method," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 6, pp. 3993–4005, 2016.

[23] R. A. Sivakumar and R. Batta, "The variance-constrained shortest path problem," *Transportation Science*, vol. 28, no. 4, pp. 309–316, 1994.

[24] H. Hu and R. Sotirov, "Special cases of the quadratic shortest path problem," *Journal of Combinatorial Optimization*, vol. 35, no. 3, pp. 754–777, 2018.

[25] L. Yang and X. Zhou, "Optimizing on-time arrival probability and percentile travel time for elementary path finding in time-dependent transportation networks: Linear mixed integer programming reformulations," *Transportation Research, Series B: Methodological*, vol. 96, pp. 68–91, 2017.

[26] W. Zeng, T. Miwa, Y. Wakita, and T. Morikawa, "Application of lagrangian relaxation approach to α-reliable path finding in stochastic networks with correlated link travel times," *Transportation Research Part C: Emerging Technologies*, vol. 56, pp. 309 – 334, 2015.

[27] T. Xing and X. Zhou, "Finding the most reliable path with and without link travel time correlation: A lagrangian substitution based approach," *Transportation Research, Series B: Methodological*, vol. 45, no. 10, pp. 1660–1679, 2011.

[28] Y. Zhang and A. Khani, "An algorithm for reliable shortest path problem with travel time correlations," *Transportation Research Part B: Methodological*, vol. 121, pp. 92 – 113, 2019.

[29] Y. Gao, "Shortest path problem with uncertain arc lengths," *Computers & Mathematics with Applications*, vol. 62, no. 6, pp. 2591 – 2600, 2011.

[30] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, 1959.

[31] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *Pervasive Computing, IEEE*, vol. 7, no. 4, pp. 12–18, 2008.