2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Incremental Sampling-based Algorithm for Risk-aware Planning under Motion Uncertainty

Wei Liu and Marcelo H. Ang Jr.

Abstract— **This paper considers the problem of motion planning for linear systems subject to Gaussian motion noise and proposes a risk-aware planning algorithm: CC-RRT*-D. The proposed CC-RRT*-D employs the chance-constraint approximation and leverages the asymptotically optimal property of RRT* framework to compute risk-aware and asymptotically optimal trajectories. By explicitly considering the state dependence for prior state estimate, the over-conservative problem of chance-constraint approximation can be provably solved. Computational experiment results show that CC-RRT*-D is efficient and robust compared with related algorithms. The real-time experiment on an autonomous vehicle shows that our proposed algorithm is applicable to real-time obstacle avoidance.**

I. Introduction

Recent advances in autonomous vehicle research have greatly improved the vehicles' performance and safety. Autonomous navigation in clustered urban environment, however, still faces the challenges of the uncertainties that arising from either external sources (like moving agents' intention, etc.) or internal source (like model errors, control noise, etc.). Recognizing these challenges, a motion planner that is robust to the uncertainty is in need for autonomous vehicles to perform safe and complex maneuvers in a timely manner.

The problem of motion planning with control disturbance, sensing noise and moving agents' dynamics was attracting increasing attention recently. These uncertainties make the vehicle operate in partially observable space (belief space), where the environment and vehicle states are described as probability distributions. Some approaches tend to blend planning and control by returning a global control policy over the entire space. Markov Decision Processes (MDPs), for instance, can take account motion uncertainty and optimize the probability of success [1]. However, the discretization of the state and control space is required. In order to take account sensing uncertainty, the MDP concept can be extended to Partially Observable Markov Decision Processes (POMDPs) [3], which is, however, computationally intractable for realistic problems.

For systems with Linear dynamic, Quadratic cost and Gaussian noise (LQG), the optimal policy can be obtained via employing Kalman filter to maintain a Gaussian state estimate and evaluate the performance metric over the estimate [4]. Blackmore $et\ al.$ presented a chance-constraint approximation approach in [5] to compute probabilistically robust trajectories under Gaussian disturbance. To get rid of the LQG limitation, this work was further improved in [6] to handle non-linear systems with non-Gaussian disturbance. In order to compute trajectories with smaller sub-optimality, Ono $et\ al.$ in [7] presented the Iterative Risk Allocation (IRA) approach by intelligently allocating risk limit for each constraint. While these approaches have been demonstrated for real-time planning, they lack the adaptiveness to complex and high-dimension problem.

Recognizing the recent research improvements obtained in sampling-based planning algorithm for deterministic systems, many algorithms have been proposed to extend sampling-based algorithms to stochastic systems and leverage the benefits of 1) easier scalability to the high-dimension space, and 2) increasing planning horizon due to the piecewise path property. Particle-RRT is a Rapidly-exploring Random Tree (RRT) based approach which uses particles for distribution propagation, and focuses on the particle clustering and tree expansion strategy [8]. A sampling strategy for Probabilistic RoadMap (PRM) to handle environmental uncertainty is presented in [9]. The Belief Roadmap in [10] performs an efficient belief space planning using factorized covariance for linear or locally linearizable systems. Luders $et\ al.$ proposed a Chance-Constraint RRT (CC-RRT) in [11] to build RRT tree with a bounded collision probability. While improvements achieved, these approaches can only return sub-optimal trajectories due to the non-optimality of RRT and PRM that proved in [12].

With the emergence of the asymptotically optimal planning algorithms (like RRG, RRT*, PRM*) in [12], the Rapidly-exploring Random Belief Tree (RRBT) algorithm proposed in [13] aims at constructing and refining a belief tree within the RRT* framework to handle motion and sensing uncertainty. Recent work in [14] improves the Chance-Constraint RRT by using RRT* instead. Both these two approaches, however, failed to consider the state independence when prior state distribution is estimated and RRT* tree is rewired.

In this paper, we propose a risk-aware planning algorithm CC-RRT*-D for linear or locally linearizable systems under Gaussian motion disturbance. The proposed CC-RRT*-D explicitly considers the state dependence for chance-constraint approximation and integrate it into the RRT* framework. Compared to the previous work, the contribution of this paper can be summarized as follows:

- State dependence for state distribution estimate is explicitly considered, which can provably overcome the

Wei Liu and Marcelo H. Ang Jr. are with Department of Mechanical Engineering, National University of Singapore, Singapore {liu_wei, mpeang} @ nus.edu.sg

*This research was supported by the Future Urban Mobility project of the Singapore-MIT Alliance for Research and Technology (SMART) Center, with funding from Singapore's National Research Foundation.

over-conservativeness introduced by chance-constraint approximation.
- Risk-awareness and asymptotically optimality can be guaranteed by integrating chance-constraint approximation into RRT* framework.

The paper proceeds as follows: Section II formulates the planning problem. In Section III, chance-constraint approximation, conditional state propagation and the proposed planning algorithm are discussed step by step. The simulation results and a real-time experiment are demonstrated in Section IV. Finally, this paper is concluded in Section V.

II. Problem Statement

Let the $\mathcal{X} \subset \mathbb{R}^{n_x}$ be the state space, and let $\mathcal{U} \subset \mathbb{R}^{n_u}$ be the control space. We assume that applying a control input $u_t \in \mathcal{U}$ at stage t brings the vehicle from state x_t to state x_{t+1} according to the following stochastic model:

$$\begin{aligned} x_{t+1} &= f(x_t, u_t, m_t), \\ m_t \sim \mathcal{N}(0, M_t),\ & x_{init} \sim \mathcal{N}(\hat{x}_0, \Sigma_{x_0}), \end{aligned} \tag{1}$$

where $m_t \in \mathbb{R}^{n_u}$ is the motion noise that imposed on the control input. x_{init} is the initial state and is assumed as a Gaussian distribution with mean and covariance as $\hat{x}_0$ and Σ_{x_0} respectively. The transition function f is assumed to be either linear or locally well approximated by its linearization.

Let the obstacle free space be represented as $\mathcal{X}_F$, which can take the form as:

$$\mathcal{X}_F = \mathcal{X} - \mathcal{X}_1 - \ldots - \mathcal{X}_K, \tag{2}$$

where $\mathcal{X}_j, j \in \{1, ..., K\}$ is convex polyhedral that models the obstacle regions, and operator $'-'$ denotes set subtraction.

Let $\mathcal{X}_{goal}$ denote the goal region, and let π denote a path that defined as a series of states and control inputs $(x_0, u_0, ..., x_{T_{goal}}, u_{T_{goal}})$ such that $x_0 = x_{init}$ and $x_{T_{goal}} \in \mathcal{X}_{goal}$. Then a path planner with motion uncertainty seeks to solve the optimal control problem:

$$\begin{aligned} &\min_{u_t} \sum_{t=0}^{T_{goal}} \phi(\hat{x}_t, \mathcal{X}_{goal}, u_t), \\ &\text{subject to Eqn. (1)}, \\ &\text{and } P(x_t \in \mathcal{X}_F) \geq \delta, t : [0, T_{goal}], \end{aligned} \tag{3}$$

where ϕ is the object function to be optimized and $\hat{x}_t$ denotes the expected mean value of vehicle's state distribution at stage t. The usage of $\hat{x}_t$ here is to approximate the expected cost that associated with x_t. δ is the probabilistic safety limit and $P(x_t \in \mathcal{X}_F) \geq \delta$ is to guarantee the bounded collision probability.

III. Algorithm

This section describes the risk-aware and asymptotically optimal planning algorithm for system under motion uncertainty.

A. Chance Constraint

Given a convex polyhedral obstacle $\mathcal{X}_o$ that defined by L linear segments, the event that vehicle collides with obstacle $\mathcal{X}_o$ at time t can be modeled as a conjunction of linear constraints on the vehicle state x_t:

$$\bigwedge_{i=1,\ldots,L} a_i^T x_t < b_i, \tag{4}$$

where a_i and b_i are the parameters that model the ith linear constraint.

If x_t is given as a distribution estimate at stage t, the collision with the obstacle can be evaluated as the probability of Eqn. (4) being satisfied. Recognizing the obstacle is assumed as convex, thus the probability that any of the linear constraints in (4) being satisfied is an upper bound on the obstacle colliding probability:

$$P(x_t \in \mathcal{X}_o) = P(\bigwedge_{i=1,\ldots,L} a_i^T x_t < b_i) \leq P(a_i^T x_t < b_i). \tag{5}$$

Assume x_t is a Gaussian distribution defined by a mean μ_t and covariance Σ_t, and define an affine transform as $V_{it} = a_i^T x_t - b_i$. The probability that the ith linear constraint is satisfied can be calculated as:

$$P(V_{it} < 0) = [1 - erf((a_i^T \mu_t - b_i)/\sqrt{2a_i^T \Sigma_t a_i})]/2, \tag{6}$$

where erf denotes the standard error function. This result can be inserted into the usage of Boole's inequality in Eqn. (5) to give a approximated evaluation of the obstacle colliding probability:

$$P(x_t \in \mathcal{X}_o) = \min_{i \in \{1, \ldots L\}} P(V_{it} < 0). \tag{7}$$

Considering the case of multiple obstacles and let $P_t^j = P(x_t \in \mathcal{X}_j), j \in \{1, \ldots, K\}$ denotes the probability of collision with jth obstacle, we loosely define the overall collision probability as:

$$P(x_t \notin \mathcal{X}_F) = \max_{j \in \{1, \ldots, K\}} P_t^j, \tag{8}$$

which might not reflect the true risk, but provides a conservative approximation by considering the worst case.

B. Conditional State Propagation

In principle, our proposed approach applies to linear dynamics. Considering the fact that the vehicle will be controlled to stay close to nominal path during execution, the non-linear models can be locally linearized around the nominal path as:

$$\bar{x}_t = A_t \bar{x}_{t-1} + B_t \bar{u}_t + V_t m_t, \tag{9}$$

where

$$\bar{x}_t = x_t - x_t^*, \bar{u}_t = u_t - u_t^* \tag{10}$$

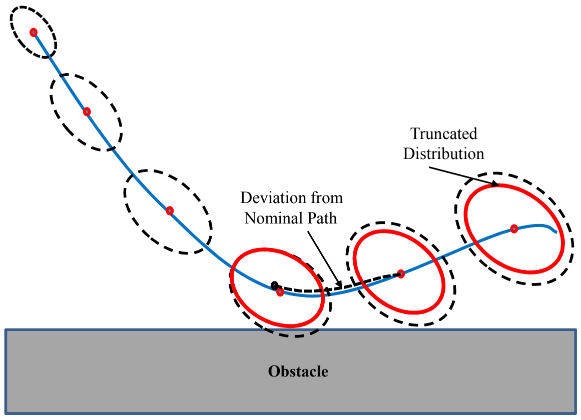

Fig. 1. Illustration of conditional state propagation: conditional state propagation based state distribution is represented as red ellipses, and the black dashed ellipses denote the state distribution without using conditional state propagation.

are the deviation from nominal state x_t^* and control u_t^* respectively, and

$$A_t = \frac{\partial f}{\partial x}(x_{t-1}^*, u_{t-1}^*, 0), \quad B_t = \frac{\partial f}{\partial u}(x_{t-1}^*, u_{t-1}^*, 0), \quad V_t = \frac{\partial f}{\partial m}(x_{t-1}^*, u_{t-1}^*, 0) \tag{11}$$

are the Jacobian matrices of f along nominal path.

Since the true state x_t as well as $\bar{x}_t$ are not fully observable during the execution, the Kalman filter can be employed to approximate the state estimate as Gaussian distribution and keep track the estimate mean $\hat{\bar{x}}_t$ and the covariance Σ_t of $\bar{x}_t$. Previous works use the maximum likelihood measurement model to update the prediction, which cannot reflect the true state distributions, but rather a measure of how well one will be able to be inferred. Hence the estimate of $\bar{x}_t$ is given as following without measurement update:

$$\hat{\bar{x}}_t = A_t\hat{\bar{x}}_{t-1} + B_t\bar{u}_t, \quad \Sigma_t = A_t\Sigma_{t-1}A_t^T + V_tM_tV_t^T. \tag{12}$$

Then the estimate of x_t can be given as: $\tilde{x}_t \sim \mathcal{N}(\hat{\bar{x}}_t + x_t^*, \Sigma_t)$, and $\hat{x}_t = \hat{\bar{x}}_t + x_t^*$ denotes the mean of $\tilde{x}_t$.

Given a feedback control system: $\bar{u}_t = K_t\hat{\bar{x}}_t$, then $\hat{\bar{x}}_t$ is formulated as $\hat{\bar{x}}_t = \prod_{\tau=0:t-1}(A_\tau + B_\tau K_\tau)\hat{\bar{x}}_\tau$. Noticed that $\hat{\bar{x}}_0 = 0$, thus $\hat{\bar{x}}_t$ can be controlled as 0, i.e., the expected mean of the prior distribution is staying exactly in the nominal path.

Considering the dependency between state x_t and x_{t-1} in Eqn. (1), the feasibility of x_t should be strictly conditional on the previous states being collision free, i.e., $P(x_t \in \mathcal{X}_F) = P(x_t \in \mathcal{X}_F \mid \bigwedge_{i=0:t-1} x_i \in \mathcal{X}_F)$. In this sense, we let

$$\tilde{x}_{t|k} = (\tilde{x}_t \mid \bigwedge_{i=0:k} \tilde{x}_i \in \mathcal{X}_F) \tag{13}$$

denotes the state distribution of $\tilde{x}_t$ that conditional on the state being collision free for all stages $\{0, ..., k\}$, and utilize the method proposed in [15] to approximate the distribution

Algorithm 1: Conditional State Propagation

```
input : $\tilde{x}_{t-1|t-1} \sim \mathcal{N}(\hat{x}_{t-1|t-1}, \Sigma_{t-1|t-1})$
output: $\tilde{x}_{t|t}$
1  $\hat{\bar{x}}_{t-1|t-1} = \hat{x}_{t-1|t-1} - x_{t-1}^*$
2  $\hat{\bar{x}}_{t|t-1} = A_t\hat{\bar{x}}_{t-1|t-1} + B_t(\bar{u}_t)$
3  $\Sigma_{t|t-1} = A_t\Sigma_{t-1|t-1}A_t^T + V_tM_tV_t^T$
4  $\hat{x}_{t|t-1} = \hat{\bar{x}}_{t|t-1} + x_t^*$
5  $\tilde{x}_{t|t-1} \sim \mathcal{N}(\hat{x}_{t|t-1}, \Sigma_{t|t-1})$
6  if DegreeTrunc($\tilde{x}_{t|t-1}$) $> \omega$ then
7      $[\Delta x_t, \Delta\Sigma_t] \leftarrow$ Truncation($\tilde{x}_{t|t-1}$)
8      $\hat{x}_{t|t} = \hat{x}_{t|t-1} - \Delta x_t$
9      $\Sigma_{t|t} = \Sigma_{t|t-1} - \Delta\Sigma_t$
10     return $\tilde{x}_{t|t} \sim \mathcal{N}(\hat{x}_{t|t}, \Sigma_{t|t})$
11 else
12     return $\tilde{x}_{t|t} \sim \mathcal{N}(\hat{x}_{t|t-1}, \Sigma_{t|t-1})$
13 end
```

$\tilde{x}_{t|t} \sim \mathcal{N}(\hat{x}_{t|t}, \Sigma_{t|t})$ as all collision-free states at stage t. As illustrated in Fig. 1, this can be done by truncating the distribution $\mathcal{N}(\hat{x}_{t|t-1}, \Sigma_{t|t-1})$ against the infeasible region in the environment and results in a shift of the mean and covariance by Δx_t and $\Delta\Sigma_t$ respectively:

$$\hat{x}_{t|t} = \hat{x}_{t|t-1} - \Delta x_t, \quad \Sigma_{t|t} = \Sigma_{t|t-1} - \Delta\Sigma_t. \tag{14}$$

Given a linear constraint $a^T\tilde{x}_{t|t-1} \leq b$, the shift Δx_t and Σ_t can be given as:

$$\Delta x_t = -\frac{\lambda}{\sqrt{a^T\Sigma_{t|t-1}a}}(\Sigma_{t|t-1}a), \quad \Delta\Sigma_t = \frac{\lambda^2 - \beta\lambda}{a^T\Sigma_{t|t-1}a}(\Sigma_{t|t-1}a)(\Sigma_{t|t-1}a)^T, \tag{15}$$

where $\beta = \frac{b - a^T\hat{x}_{t|t-1}}{\sqrt{a^T\Sigma_{t|t-1}a}}$ is the degree of truncation, and $\lambda = \frac{pdf(\beta)}{1-cdf(\beta)}$ is the ratio of the standard Gaussian probability distribution function and the cumulative distribution function evaluated at β.

Let us consider the distribution truncation w.r.t. obstacles that defined by L linear constraints. Because of the approximation made in Eqn. (7), the truncation is only applied to the linear constraint that returns the minimum risk, which can be formulated as:

$$[a, b] = \arg\min_{[a_i, b_i]} P(a_i\tilde{x}_{t|t-1} < b_i), i \in \{1, \ldots, L\}. \tag{16}$$

Given K obstacles, let $[\Delta x_t^j, \Sigma_t^j]$ be the shift pair for jth obstacle $\mathcal{X}_j$, then the truncation w.r.t. all the obstacles is given as the cumulative shift:

$$\Delta x_t = \Sigma_{j=0}^K \Delta x_t^j, \; \Delta\Sigma_t = \Sigma_{j=0}^K \Delta\Sigma_t^j. \tag{17}$$

Then the conditional state propagation is summarized as Alg. 1. At each stage t of conditional state propagation, the prior distribution $\tilde{x}_{t|t-1}$ is truncated if the degree of truncation returned by function DegreeTrunc is over the desired

Algorithm 2: CC-RRT*-D Tree Expansion

```
input : $x_{init}$
output: $T = (V, E)$
1  $T \leftarrow$ InitializeTree()
2  $T \leftarrow$ InsertNode($\emptyset, x_{init}, T$)
3  for $i \leftarrow 1$ to $N$ do
4      $x_{rand} \leftarrow$ Sample()
5      $x_{nearest} \leftarrow$ Nearest($T, x_{rand}$)
6      $(x_{new}, u_{new}, \pi_{new}) \leftarrow$ Steer($x_{nearest}, x_{rand}$)
7      if RiskFeasible($x_{nearest}, u_{new}, x_{new}$) then
8          $X_{near} \leftarrow$ Near($T, x_{new}, \eta$)
9          $x_{min} \leftarrow$ Parent($X_{near}, x_{new}$)
10         $T \leftarrow$ InsertNode($x_{min}, x_{new}, \pi$)
11         $T \leftarrow$ Rewire($T, X_{near}, x_{min}, x_{new}$)
12     end
13 end
14 return $T = (V, E)$
```

Algorithm 3: $RiskFeasible(x_{start}, u, x_{end})$

```
input : $x_{start}, u, x_{end}$
output: $False, True$
1  $\mathcal{N}(\hat{x}_{0|0}, \Sigma_{0|0}) \leftarrow x_{start}$
2  for $t \leftarrow 1$ to $k$ do
3      $\tilde{x}_{t|t-1} \leftarrow$ Propagate($\mathcal{N}(\hat{x}_{t-1|t-1}, \Sigma_{t-1|t-1})$)
4      if $\mathbb{P}(Collision) > 1 - \delta$ then
5          return $False$
6      end
7      $\mathcal{N}(\hat{x}_{t|t}, \Sigma_{t|t}) \leftarrow$ Truncate($\tilde{x}_{t|t-1}$)
8  end
9  $x_{end} \leftarrow \mathcal{N}(\hat{x}_{k|k}, \Sigma_{k|k})$
10 return $True$
```

Algorithm 4: $Parent(X_{near}, x_{new})$

```
input : $X_{near}, x_{new}$
output: $x_{min}, J_{min}$
1  for $x \in X_{near}$ do
2      $(x', w', \pi') \leftarrow$ Steer($x, x_{new}$)
3      if RiskFeasible($x', w', x_{new}$) then
4          $X_{steer} \leftarrow$ InsertVertex($x$)
5      end
6  end
7  $J_{min} = \min_{x \in X_{steer}} J(x) + c(x, x_{new})$
8  $x_{min} = \arg\min_{x \in X_{steer}} J(x) + c(x, x_{new})$
```

Algorithm 5: $Rewire(T, X_{near}, x_{min}, x_{new})$

```
input : $T, X_{near}, x_{min}, x_{new}$
output: $T$
1  for $x \in X_{near} \backslash x_{min}$ do
2      $(x', w', \pi') \leftarrow$ Steer($x_{new}, x$)
3      if RiskFeasible($x_{new}, w', x'$) and
       $J(x_{new}) + c(x_{new}, x) < J(x)$ then
4          $T \leftarrow$ Reconnect($x_{new}, x, \pi$)
5          RePropagate($x$)
6      end
7  end
```

threshold ω, then the truncated distribution $\mathcal{N}(\hat{x}_{t|t}, \Sigma_{t|t})$ is utilized for the propagation of next stage $t+1$. Because of the mean shift Δx_t, using $\mathcal{N}(\hat{x}_{t|t}, \Sigma_{t|t})$ to propagate next stage will introduce some deviation from the nominal path (Black dashed curve in Fig. 1), which is assumed negligible when the horizon between stage t and stage $t+1$ is small enough.

Consequently, the probability of a path which consisting of states $x_{t:0,\ldots,k}$ being collision free is approximated as:

$$P(\bigwedge_{t=0}^{k} x_t \in \mathcal{X}_F) \approx \prod_{t=0}^{k} P(\tilde{x}_{t|t-1} \in \mathcal{X}_F \mid \bigwedge_{i=0:t-1} \tilde{x}_{i|i-1} \in \mathcal{X}_F), \tag{18}$$

which is less conservative than the approximation $P(\bigwedge_{t=0}^{k} x_t \in \mathcal{X}_F) \approx \prod_{t=0}^{k} P(\tilde{x}_t \in \mathcal{X}_F)$ that given by some previous approaches.

C. CC-RRT*-D Algorithm

The tree expansion algorithm is given as Alg. 2. In each iteration, the $Sample$ function generates independent, identically distributed samples over the feasible state space for expansion. Then the nearest node in terms of some distance metric is identified, and the $Steer$ function is applied to computes a path π_{new} that connecting the nearest node with a intermediate state x_{new}. The function $RiskFeasible$ in Alg. 3 acts as the collision checker, which employs Alg. 1 to propagate the states' distribution estimate along the nominal path, and check the probability of collision using Eqn. (8), which will return true if the given trajectory is probabilistically safe. If x_{new} and π_{new} are probabilistically feasible, a near vertex set X_{new} will be returned by the function $Near(T, x, \eta)$ as:

$$X_{near} = \{x' \mid ||x' - x|| \leq \min\{\gamma(log(n)/n)^{1/d}, \eta\}, \tag{19}$$

where d is the dimension of the state, n is the number of vertex in the tree, η is a pre-defined maximum ball radius and γ is the constant that discussed in [12].

For all the near vertex that has probabilistically feasible connection with x_{new}, the best one w.r.t. the cost function c is set as the parent of x_{new} (Alg. 4) and is inserted into the tree. Then $Rewire$ function in Alg. 5 recalculates the best parents for all vertex within X_{near}. Once the rewiring procedure is necessary, the tree is reconnected. Then the rewired vertex together with its children's distribution is re-propagated in order to maintain the dependency between the vertices.

Finally, Alg. 2 builds the probabilistically feasible tree and the solution path is asymptomatically optimized.

D. Analysis

1) Cost Function: Given two connected paths $\pi_1 : [0,1]$, $\pi_2 : [0,1]$ and $\pi_1(1) = \pi_2(0)$, and let $\pi_1 \mid \pi_2$ denote their

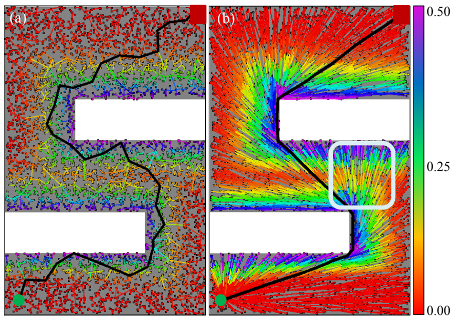

Fig. 2. Results for planning without motion noise after 5,000 vertex trials: a) RRT and b) RRT*.

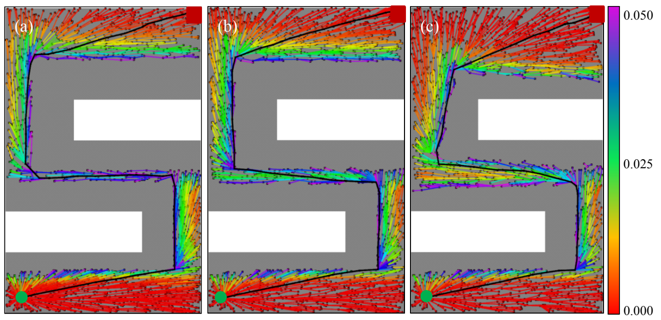

Fig. 3. Comparison of probabilistically feasible trees of three algorithms over 2,000 vertex trials: a) CC-RRT*, b) CC-RRT*-R and c) CC-RRT*-D.

concatenation. To ensure the optimality, the cost function c in Alg. 2 needs to be monotonic, in sense of $c(\pi_1) \leq c(\pi_1 \mid \pi_2)$, and bounded, in sense that there exists κ such that $c(\pi) \leq \kappa||\pi||$ [12].

Let $R(x)$ denote the risk of collision that associated with state x. The line integral of the risks along a given path $\pi : [0,1]$ as $\int_0^1 R(\pi(\tau))d\tau$ only provides the accumulated risk, which, however, cannot reflect the significant risky component of path π. Therefore, the maximum risk over the whole path is employed to evaluate the path's risky level, and a risk based object function can be defined as:

$$c(\pi) = Length(\pi) + \rho \max_{\tau:[0,1]} R(\pi(\tau)), \tag{20}$$

where ρ is the scaling factor to balance the path length and risk. Because both the $Length(\pi)$ and $\max_{\tau:[0,1]} R(\pi(\tau))$ are monotonic and bounded, this cost function can be easily verified to be able to meet the requirement of optimality.

2) Optimality Analysis: To prove the optimality of Alg. 2, we start from a necessary assumption.

$Assumption$: There exists a ball $\mathcal{X}^\gamma$ of radius γ at every point $x \in \mathcal{X}$ such that for points $x' \in \mathcal{X}^\gamma$, $\int_{\mathcal{X}_F} P(x')dx' > \delta$.

This assumption states that it is possible to move the mean of the distribution within some ball and not violate the chance-constraint, which is necessary to give the graph a finite sample volume to converge in. Based on this assumption, the optimality proof can follow the Theorem 38 in [12].

IV. Experiment

In this section, we will present the simulation results to evaluate the performance of the CC-RRT*-D, and demonstrate a real-time experiment on an autonomous golf-cart.

A. Computation Experiment

1) Simulation Setup: The computational experiments were implemented on a quad-core 1.6 GHz processor with 4 GB of RAM, and the algorithms are programmed in C++.

Consider the 2D single integrator dynamics:

$$x_t = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_{t-1} + \begin{bmatrix} dt & 0 \\ 0 & dt \end{bmatrix} [v_x + \Delta v_x, v_y + \Delta v_y]^T, \tag{21}$$

where $dt = 0.1\ s$ and $|v_x| \leq 10\ m/s, |v_y| \leq 10\ m/s$. The motion noise is imposed on the input velocity as $[\Delta v_x, \Delta v_y]^T$.

The simulation was conducted in a constrained, two dimension 10 m × 15 m environment, which contains two obstacles as Fig. 2. The starting location is set as the left lower conner and goal region is located at the right upper corner.

The initial state x_0 is subject to a Gaussian localization error: $x_0 \sim \mathcal{N}(\hat{x}_0, \Sigma_0)$, $\Sigma_0 = \left[\begin{smallmatrix} 0.5 & 0 \\ 0 & 0.5 \end{smallmatrix}\right]$. The motion noise are given as $[\Delta v_x, \Delta v_y]^T \sim \mathcal{N}\left(0, \left[\begin{smallmatrix} 0.2|v_x| & 0 \\ 0 & 0.2|v_y| \end{smallmatrix}\right]\right)$. The probabilistic safety limit is set as 0.95, i.e., $P(collision) \leq 0.05$, and the degree of truncation threshold ω is set as 0.005.

2) Simulation Results: Fig. 2 represents the planning results without considering motion noise using RRT and RRT* after 5,000 vertex, where the path length is defined as the object function. The edge's color indicates the risky level of the corresponding piecewise path, from which we can intuitively find that risk is increasing when obstacles are approached. The region highlighted by white rectangle is associated with relatively higher risk and will be proved as a bottleneck when motion disturbance is imposed.

Then we compared the planning results of our proposed algorithm (CC-RRT*-D) with a previous related approach CC-RRT* [14] in which state dependence is not explicitly considered.

To explicitly demonstrate the difference introduced by considering state dependence, we compared three algorithms: 1) CC-RRT*, 2) CC-RRT*-R: Same framework as Alg. 2 but using Eqn. (12) for state propagation and 3) CC-RRT*-D. The comparison between CC-RRT* and CC-RRT*-R is to show the necessity of state re-propagation when rewiring procedure is proceeded. By comparing algorithm CC-RRT*-R and CC-RRT*-D, we can observe the improvements obtained in solving over-conservative issue of chance-constraint approximation.

Without the loss of generality, path length is defined as the cost function. 2,000 vertex trials were computed each time, and each algorithm was tested over 20 times. The probabilistically feasible tree for each algorithm is shown in Fig. 3.

Firstly, we explore the necessity of state re-propagation when rewiring procedure is proceeded. In Fig. 3(a), the lower

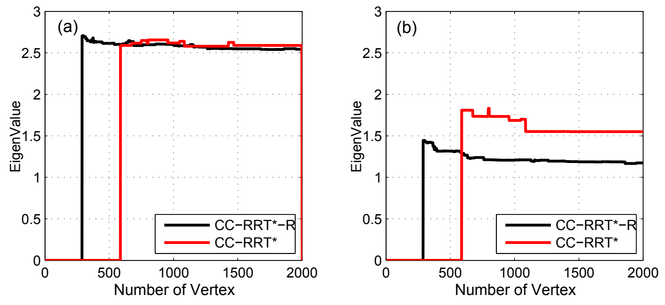

Fig. 4. Covariance matrix's eigenvalues of the solution path's end state: a) large eigenvalues and b) small eigenvalues.

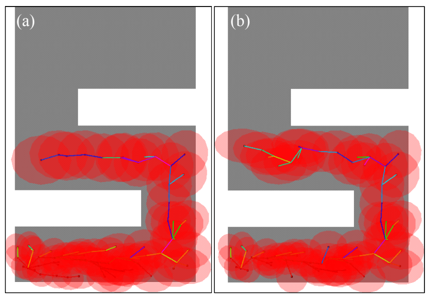

Fig. 6. State distribution estimate after 100 vertices (3σ): a) Unconditional state propagation, b) Conditional state propagation.

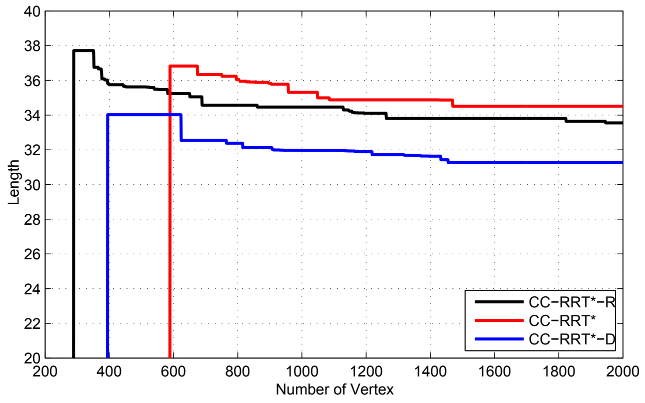

Fig. 5. Evolution of solution path's length over 2,000 vertex trials.

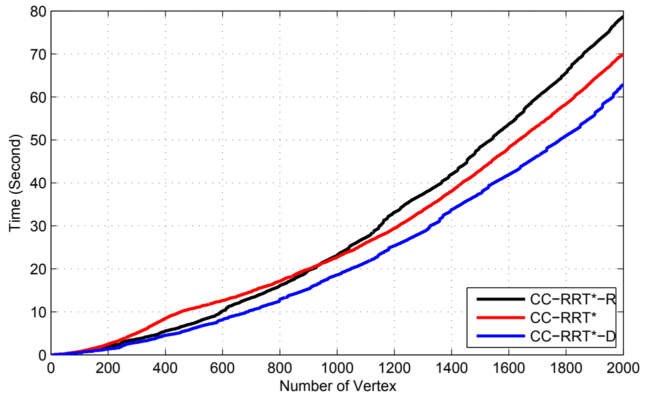

Fig. 7. Computation time versus number of vertex over 2,000 vertex trials.

part of CC-RRT* tree is relatively denser than the upper part, and is more conservative in the constrained region when comparing to CC-RRT*-R tree in Fig. 3(b). By analyzing the tree expansion process, this is because CC-RRT* need to make more efforts to find a path that is passing the bottleneck which highlighted in Fig. 2, hence more vertices are expanded in the lower part. This can also be verified by the result in Table I, where CC-RRT* use around 465 vertices to find a feasible solution but CC-RRT*-R only takes about 372 vertices.

The underlying reason is that each vertex's covariance is a function of the path that linking this vertex to the tree's root. And for this typical case, shorter path normally requires less control effort, hence less motion noise is imposed on state propagation. Therefore, along with the convergence of the vertex's distance to the root, its estimated covariance should also be converging. Let x_{goal} denotes the end state of solution path, the eigenvalues of x_{goal}'s covariance matrix are plotted in Fig. 4, from which we can view the asymptotic convergence of the covariance, and CC-RRT*-R returned smaller covariance more efficiently. This also explains why CC-RRT*-R can handle the bottleneck more easily.

TABLE I
NUMBER OF VERTICES NEEDED TO FIND FEASIBLE PATH (20 TRIALS)

	CC-RRT*	CC-RRT*-R	CC-RRT*-D
Average	465.5	372.8	388.0
Stdev.	229.0	174.1	166.65
Min.	262	186	190
Max.	958	693	690

Then we compare the CC-RRT*-R tree (Fig. 3(b)) with CC-RRT*-D tree (Fig. 3(c)). CC-RRT*-R tree is obviously more conservative than that of CC-RRT*-D, hence the solution path returned by CC-RRT*-D is relatively shorter. This is in consistence with previous discussion in Section III-B: because the infeasible states are truncated from the prior distribution, the state propagation based on the truncated distribution is relatively less conservative. The corresponding state propagation results after 100 vertices is shown in Fig. 6.

Fig. 5 charts the evolution of solution path's length versus number of vertex. The median over all 20 trials for each algorithm is plotted, where the solution paths are demonstrated as being asymptotically optimized. CC-RRT*-D returned the shortest path, and the solution path returned by CC-RRT*-R is relatively shorter than that of CC-RRT*.

The computation time versus number of vertex is shown as Fig. 7. Due to the bottleneck, CC-RRT* need more time to generate vertex initially, and the CC-RRT* tree is quickly expanded once a feasible path is available. Overall, CC-RRT* is relatively more computational efficient than CC-RRT*-R, because CC-RRT*-R consumes additional computation for state re-propagation when the tree is rewired. While additional computation is required by CC-RRT*-D for state re-propagation and distribution truncation, the less conservative property makes CC-RRT*-D take the least time to build the tree.

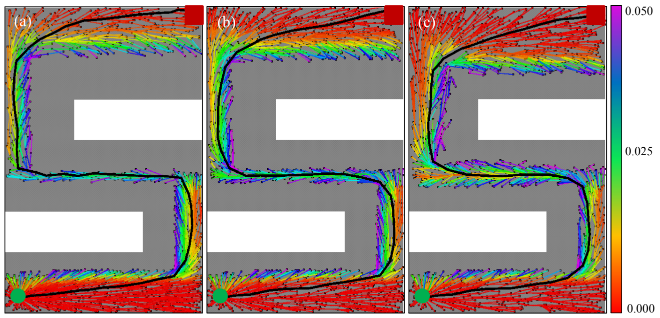

Fig. 8. Probabilistically feasible trees of three algorithms using risk based cost function over 2,000 vertex trials: a) CC-RRT*, b) CC-RRT*-R and c) CC-RRT*-D.

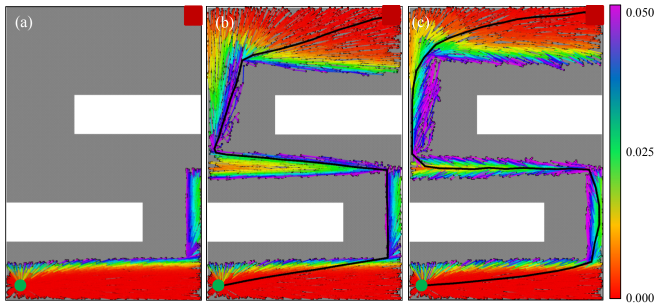

Fig. 9. Planning results using CC-RRT*-R and CC-RRT*-D under high motion noise: a) CC-RRT*-R failed to find a solution path, b) CC-RRT*-D using path length as object function, and c) CC-RRT*-D using risk based object function.

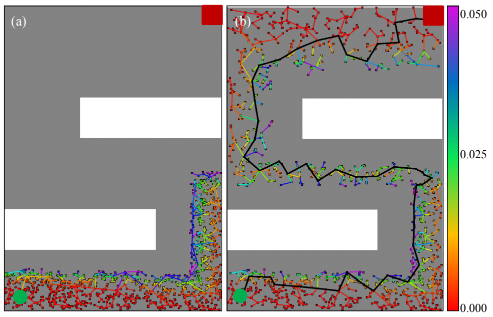

Fig. 10. Conditional state propagation extended to CC-RRT: a) CC-RRT without using conditional state propagation and b) CC-RRT using conditional state propagation.

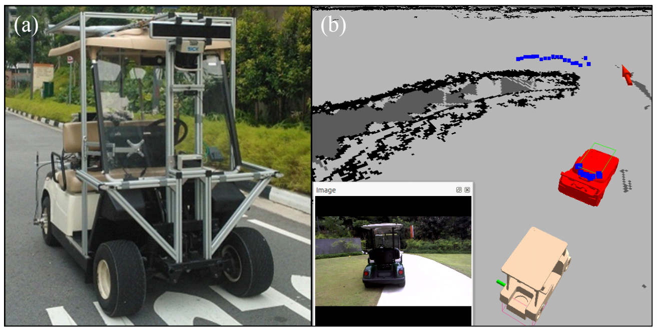

Fig. 11. Real-time experiment Setup: a) Autonomous golf-cart mounted with various sensors; b) Experiment scenario: The autonomous need to avoid the front golf-cart and navigate to the destination that marked as red arrow.

Fig. 8 represents the planning results using the risk based cost function with $\rho = 10$. The resulting trees show significant qualitative difference with that in Fig. 3. Trees in Fig. 8 tend to travel toward the obstacle surfaces to minimize the duration spend by traveling in high risk regions, and trade off between path length and risky level. In Fig. 3, however, the paths that passing around obstacles tend to be parallel with the obstacle surfaces to minimize the path length.

Then we increased the motion noise to $[\Delta v_x, \Delta v_y]^T \sim \mathcal{N}(0, \begin{bmatrix} 0.4|v_x| & 0 \\ 0 & 0.4|v_y| \end{bmatrix}$ and implement the same simulation setup over 5,000 vertex. As in Fig. 9(a), CC-RRT*-R failed to return a solution due to the over-conservativeness of chance-constraint approximation. On the other hand, solution can be efficiently found by CC-RRT*-D after about 1,500 vertex trials using two different object functions as Fig. 9(b) and Fig. 9(c).

Last but not least, the extension of the conditional state propagation to CC-RRT [11] is also implemented. Fig. 10 represents the comparison between the CC-RRT tree with and without using conditional state propagation after 1,000 vertex trials. In contrast to CC-RRT in Fig. 10(a) which fails to find solution, CC-RRT using conditional state propagation can efficiently handle the constrained region as Fig. 10(b).

To summarize, the experiment results presented in this subsection has verified the necessity of re-propagation when tree rewiring is proceeded, and the introduction of conditional state propagation can solve the over-conservativeness issue of chance-constraint approximation.

B. Real-time Experiment for Obstacle Avoidance

We implemented a real-time obstacle avoidance using our proposed CC-RRT*-D on an autonomous vehicle.

1) Experiment Setup: The experiment was conducted on an autonomous golf-cart mounted with various affordable sensors [16] (Fig. 11(a)) inside the campus of National University of Singapore.

Regarding the system model, the state of the vehicle $[x, y, \theta]$ consists of its position $[x, y]$, and its orientation θ. The control input $u = [v, \phi]$, consists of its velocity v and steering angle ϕ corrupted by motion noise $m = [\Delta v, \Delta\phi] \sim \mathcal{N}(0, M)$. Hence the stochastic dynamics model is given as:

$$f(x, u, m) = \begin{bmatrix} x + (v + \Delta v)\tau cos\theta \\ y + (v + \Delta v)\tau sin\theta \\ \theta + (v + \Delta v)\tau tan(\phi + \Delta\phi)/d \end{bmatrix}, \qquad (22)$$

where τ is the time step and d is the distance between front and rear axes.

In this experiment, $|v| \leq 1.0\ m/s$, motion noise $m \sim \mathcal{N}\left(0, \begin{bmatrix} 0.1m/s & 0 \\ 0 & 0.1rad \end{bmatrix}\right)$ and the probabilistic safety limit is set as 0.9. The object function is defined as the path length, and the localization algorithm is based on Monte Carlo Localization algorithm using a synthetic 2D LIDAR [17].

The experiment scenario is given as Fig. 11(b), the autonomous golf-cart received a mission of navigating to the exit of the parking yard (Read Arrow in the right-top of Fig. 11(b)) and gracefully avoiding the front golf-cart that occupying the road.

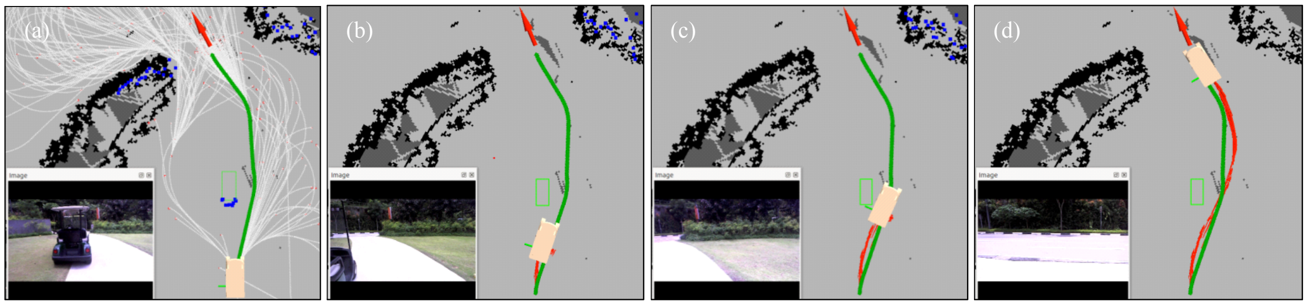

Fig. 12. Real-time experiment on an autonomous vehicle: The planning result is shown in (a) after 86 vertex trials, the solution path is indicated as green curve. The execution process is in (b)-(d), the red curve is the real trajectory that executed

2) Experiment Result: The vehicle is localized first, and the initial state distribution is approximated as Gaussian distribution by clustering the particles: $\Sigma_0 \sim \left[\begin{smallmatrix} 0.14 & 0 \\ 0 & 0.16 \end{smallmatrix}\right]$. Then the obstacle detection and classification module is called to detect the front golf-cart. Based on the obstacle classification result, the front golf-cart is modeled as a rectangle that consisting of four linear constraints, which can be visualized in Fig. 11(b).

After 86 vertex trials using 8.335 seconds, the CC-RRT*-D tree and solution path is given in Fig. 12(a). As expected, the tree is bounded in a probabilistically feasible region that imposed by chance-constraint. The solution path is then executed by low-level pure-pursuit controller, whose look-ahead distance is set as 2 meters. Fig. 12 (b)-(d) demonstrate the execution process, where the red curve indicates the real trajectory that executed. Although the solution path is not tracked closely due to the motion noise and localization error, autonomous vehicle can still safely and smoothly avoid the obstacle.

V. Conclusion

This paper has proposed the CC-RRT*-D algorithm for risk-aware and asymptotically optimal path planning. The algorithm leverages the asymptotic optimal property of RRT* framework and employs the chance-constraint to compute risk-aware and asymptotically optimal paths. The state dependence is explicitly considered in the proposed algorithm, whose necessity has been verified by the experiment results. The employment of conditional state propagation is shown to solve the over-conservative problem of previous related work. The real-time experiment in the autonomous vehicle has shown its applicability for real-time obstacle avoidance.

References

[1] S. Thrun, W. Burgard, D. Fox, *et al.*, *Probabilistic robotics*. MIT press Cambridge, 2005, vol. 1.

[2] V. A. Huynh, S. Karaman, and E. Frazzoli, "An incremental sampling-based algorithm for stochastic optimal control," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2865–2872.

[3] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1, pp. 99–134, 1998.

[4] J. Van Den Berg, P. Abbeel, and K. Goldberg, "Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[5] L. Blackmore, H. Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *American Control Conference, 2006*. IEEE, 2006, pp. 7–pp.

[6] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams, "A probabilistic particle-control approximation of chance-constrained stochastic predictive control," *Robotics, IEEE Transactions on*, vol. 26, no. 3, pp. 502–517, 2010.

[7] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*. IEEE, 2008, pp. 3427–3432.

[8] N. A. Melchior and R. Simmons, "Particle rrt for path planning with uncertainty," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 1617–1624.

[9] P. E. Missiuro and N. Roy, "Adapting probabilistic roadmaps to handle uncertain maps," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006, pp. 1261–1267.

[10] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *The International Journal of Robotics Research*, vol. 28, no. 11-12, pp. 1448–1465, 2009.

[11] B. Luders, M. Kothari, and J. P. How, "Chance constrained rrt for probabilistic robustness to environmental uncertainty," in *Proceeding of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2010.

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

[13] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 723–730.

[14] B. D. Luders, S. Karaman, and J. P. How, "Robust sampling-based motion planning with asymptotic optimality guarantees," in *AIAA Guidance, Navigation, and Control Conference (GNC), Boston, MA*, 2013.

[15] S. Patil, J. van den Berg, and R. Alterovitz, "Estimating probability of collision for safe motion planning under gaussian motion and sensing uncertainty," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3238–3244.

[16] Z. Chong, B. Qin, T. Bandyopadhyay, T. Wongpiromsarn, B. Rebsamen, P. Dai, E. Rankin, and M. Ang Jr, "Autonomy for mobility on demand," in *Intelligent Autonomous Systems 12*. Springer, 2013, pp. 671–682.

[17] Z. Chong, B. Qin, T. Bandyopadhyay, Wongpiromsarn, and M. Ang Jr, "Synthetic 2d lidar for precise vehicle localization in 3d urban environment," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013.