Robust Sampling-based Motion Planning with Asymptotic Optimality Guarantees

Robust Sampling-based Motion Planning with Asymptotic Optimality Guarantees

Brandon D. Luders*, Sertac Karaman† and Jonathan P. How‡
Aerospace Controls Laboratory
Massachusetts Institute of Technology, Cambridge, MA
luders@mit.edu, sertac@mit.edu, jhow@mit.edu

This paper presents a novel sampling-based planner, CC-RRT*, which generates robust, asymptotically optimal trajectories in real-time for linear Gaussian systems subject to process noise, localization error, and uncertain environmental constraints. CC-RRT* provides guaranteed probabilistic feasibility, both at each time step and along the entire trajectory, by using chance constraints to efficiently approximate the risk of constraint violation. This algorithm expands on existing results by utilizing the framework of RRT* to provide guarantees on asymptotic optimality of the lowest-cost probabilistically feasible path found. A novel risk-based objective function, shown to be admissible within RRT*, allows the user to trade-off between minimizing path duration and risk-averse behavior. This enables the modeling of soft risk constraints simultaneously with hard probabilistic feasibility bounds. Simulation results demonstrate that CC-RRT* can efficiently identify smooth, robust trajectories for a variety of uncertainty scenarios and dynamics.

I. Introduction

As motion planning algorithms continue to mature and become more sophisticated, a key research focus is ensuring that said algorithms are applicable to real-world scenarios, in which a system is subject to complex, dynamic, and/or uncertain constraints.[1] Trajectories generated by the motion planner must demonstrate robustness to a significant number of uncertainty sources, not only internal to the system (*e.g.,* sensing/process noise, localization error, model uncertainty), but also due to an uncertain environment. To operate in real-time, a motion planner must be able to quickly and efficiently identify feasible plans online that are robust to this uncertainty, then continue to refine them based on desired performance criteria, such as traversal time and risk aversion.

This paper presents a novel sampling-based planner, CC-RRT*, which generates robust, asymptotically optimal trajectories in real-time, subject to process noise, localization error, and dynamic and/or uncertain environmental constraints. Probabilistic feasibility is guaranteed for linear Gaussian systems by using chance constraints to ensure that the probability of constraint violation does not exceed some user-specified threshold.[2] CC-RRT* builds upon the previously-developed chance-constrained rapidly-exploring random trees (CC-RRT) algorithm, which uses the trajectory-wise constraint checking of RRT[3] to efficiently bound the risk of constraint violation online.[4] As a result,

*Ph.D. Candidate, Dept of Aeronautics and Astronautics; Member AIAA
†Charles Stark Draper Assistant Professor of Aeronautics and Astronautics, MIT; Member AIAA
‡Richard Cockburn Maclaurin Professor of Aeronautics and Astronautics, MIT; Associate Fellow AIAA

CC-RRT can quickly identify trajectories subject to both internal and environmental uncertainty, with guaranteed minimum bounds on constraint satisfaction probability at each time step.[5] This framework is expanded in this paper to consider both chance- constrained environmental boundaries and guaranteed probabilistic feasibility over entire trajectories.

While RRT provides efficient exploration of high-dimensional state spaces, dynamically feasible trajectories, and demonstrated applicability to complex motion planning applications,[6] it has also been shown to converge almost surely to non-optimal solutions.[7] This limits the ability of RRT – and thus CC-RRT, which builds upon it – to refine feasible solutions once identified, potentially leading to low-quality trajectories heavily biased on initial tree growth.

The proposed CC-RRT* algorithm instead uses the recently-developed RRT* framework[7,8] to provide guarantees on asymptotic optimality of the lowest-cost probabilistically feasible path found, by "rewiring" the tree toward lower-cost paths. The resulting real-time algorithm asymptotically converges toward minimum-length, dynamically feasible trajectories which satisfy all time-step-wise and path-wise probabilistic feasibility constraints specified, even in complex environments. Alternatively, a novel, risk-based objective function is posed which allows the user to trade-off between minimizing path duration and risk-averse behavior. This objective uses the same risk bounds computed to check the probabilistic feasibility constraints, such that no additional computation is required, and is shown to be admissible as an RRT* objective. Unlike RRT*, CC-RRT* can thus model both hard probabilistic feasibility constraints, and soft probabilistic feasibility constraints via the risk- based objective.

After considering related work in Section II, Section III provides the problem statement. The chance constraint formulation and risk bound evaluation used by CC-RRT* is reviewed and expanded in Section IV. The CC-RRT* algorithm is introduced in Section V, while Section VI analyzes theoretical properties of the algorithm and the proposed risk-based objective function. Finally, Section VII gives simulation results demonstrating that CC-RRT* can efficiently identify smooth, robust trajectories for a variety of uncertainty scenarios and dynamics.

II. Related Work

This work falls into the larger category of planning under uncertainty, for which many algorithms have been proposed.[9] Within motion planning, types of uncertainty are often classified based on whether the motion model or environment is uncertain, and whether it concerns its present state or future predictability.[10] This work focuses on uncertainty in model predictability and environmental state, though it also admits environmental predictability uncertainty.[5]

Much work in chance-constrained path planning has focused on optimization-based frameworks. Blackmore *et al.* use Boole's inequality to obtain probabilistic feasibility for linear Gaussian systems subject to process noise and localization error,[2] while Ono and Williams use iterative risk allocation to assign risk to each constraint as part of an iterative, two-stage optimization.[11] Later work by both focuses on less conservative chance constraint evaluation for convex[12] and non-convex[13] formulations. Vitus and Tomlin develop a hybrid analytic/sampling formulation for linear Gaussian systems additionally subject to polyhedral state constraints with uncertain parameters.[14] While such optimizations have been demonstrated for real-time path planning, they lack the scalability with respect to problem complexity inherent to sampling-based algorithms, a crucial consideration in complex and dynamic environments. Because sampling-based algorithms such as CC-RRT* perform trajectory-wise constraint checking, they can avoid these scalability concerns.

Several robust motion planning algorithms have been proposed using probabilistic roadmaps (PRM),[15] which construct a multi-query graph by connecting random configuration space samples, such as for uncertain obstacle vertices[16,17] or sensing uncertainty.[18] The stochastic motion roadmap[19] constructs a Markov decision process by sampling uncertain motions from PRM nodes,

but requires discretization of inputs; later work by Huynh *et al.* removes this restriction and provides asymptotic optimality guarantees.[20] The belief roadmap[21] performs efficient belief space planning using a factored covariance for nonlinear systems subject to process and sensing noise, but is limited to kinematic motion planning. Recent work considers feedback policies for belief space planning.[22]

Rapidly-exploring random trees have been previously used for robust planning, though many existing approaches focus on internal uncertainty. Particle RRT[23] uses particles to sample uncertain motion, while Kewlani *et al.* use a finite-series approximation of the uncertainty propagation.[24] Pepy *et al.* outer-approximate uncertainty sets to guarantee robust feasibility for nonlinear systems subject to bounded internal uncertainty.[25] The LQG-MP algorithm[26] linearizes nonlinear dynamics subject to motion and sensing uncertainty, applying LQG to connect states within an RRT. Environmental uncertainty is considered on a limited basis, by numerically integrating the probability of collision between radial obstacles in a multi-agent scenario. Heuristics are used to assess path quality, such as minimizing the covariance trace or maximizing standard deviations to a collision. However, unlike CC-RRT*, the algorithm does not provide guaranteed bounds on probabilistic feasibility or user-tunable robustness parameters.

The rapidly-exploring random belief tree (RRBT)[27] samples and refines trajectories within the RRT* framework, with robustness to motion and sensing uncertainty. In RRBT, tree rewiring is performed only if the replacement trajectory achieves strictly lower cost and reduced system covariance. However, this algorithm assumes a perfectly known environment; since CC-RRT* considers uncertainties in both the system and the obstacles, it generally cannot rewire in this manner. Instead, CC-RRT* folds bounds on the risk of constraint violation – which incorporates both forms of uncertainty – directly into the cost function for rewiring. Like LQG-MP, RRBT admits nonlinear dynamics, but their subsequent linearization implies that theoretical guarantees of probabilistic feasibility can only be approximated. The CC-RRT* algorithm provides guaranteed probabilistic feasibility to multiple forms of uncertainty for linear Gaussian systems, while using efficient risk evaluation to ensure its suitability for real-time, online planning.

III. Problem Statement

Consider the linear time-invariant (LTI) discrete-time system dynamics subject to process noise

$$x_{t+1} = Ax_t + Bu_t + Gw_t, \tag{1}$$
$$w_t \sim \mathcal{N}(0, P_w), \tag{2}$$

where $x_t \in \mathbb{R}^{n_x}$ is the state vector, $u_t \in \mathbb{R}^{n_u}$ is the input vector, $w_t \in \mathbb{R}^{n_w}$ is a process noise uncertainty acting on the system, and A, B, G are matrices of suitable dimension. The disturbance w_t is unknown at current and future time steps, but has a known unbounded probability distribution (2), where $\mathcal{N}(\widehat{a}, P_a)$ represents a Gaussian random variable with mean $\widehat{a}$ and covariance P_a. The disturbances w_t are independent and identically distributed across all time steps. The initial/current state x_0 may be assumed to be either perfectly known or uncertain with known probability distribution

$$x_0 \sim \mathcal{N}(\widehat{x}_0, P_{x_0}). \tag{3}$$

The system is additionally subject to constraints acting on the system state and input. These constraints are assumed to take the form

$$x_t \in \mathcal{X}_t \equiv \mathcal{X} - \mathcal{X}_{1t} - \cdots - \mathcal{X}_{n_o t}, \tag{4}$$
$$u_t \in \mathcal{U}, \tag{5}$$

where $\mathcal{X}, \mathcal{X}_{1t}, \ldots, \mathcal{X}_{n_o t} \subset \mathbb{R}^{n_x}$ are convex polytopes, $\mathcal{U} \subset \mathbb{R}^{n_u}$, and the $-$ operator denotes set subtraction. The sets $\mathcal{X}$ and $\mathcal{U}$ define a set of time-invariant convex constraints acting on the state and input, respectively. The sets $\mathcal{X}_{1t}, \ldots, \mathcal{X}_{n_o t}$ represent n_o convex, polytopic obstacles to be avoided. The time dependence of $\mathcal{X}_t$ in (4) allows the inclusion of both static and dynamic obstacles. For each obstacle, the shape and orientation are assumed to be known, while the placement is uncertain. This is represented as

$$\mathcal{X}_{jt} = \mathcal{X}_j^0 + c_{jt}, \quad \forall\, j \in \mathbb{Z}_{1,n_o}, \tag{6}$$

$$c_{jt} \sim \mathcal{N}(\widehat{c}_{jt}, P_{c_{jt}}), \quad \forall\, j \in \mathbb{Z}_{1,n_o}, \tag{7}$$

where the $+$ operator denotes set translation and $\mathbb{Z}_{a,b}$ represents the set of integers between a and b inclusive. In this model, for the jth obstacle, $\mathcal{X}_j^0 \subset \mathbb{R}^{n_x}$ is a convex polytope of known, fixed shape, while $c_{jt} \in \mathbb{R}^{n_x}$ represents an uncertain and/or time-varying translation.

The primary objective of the motion planning problem is to reach some goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^{n_x}$ while ensuring the input constraints (5) are satisfied, while the state constraints (4) are *probabilistically* satisfied. This is represented via two types of chance constraints,

$$\mathbb{P}\left(x_t \in \mathcal{X}_t\right) \geq \delta_s, \quad \forall\, t, \tag{8}$$

$$\mathbb{P}\left(\bigwedge_t x_t \in \mathcal{X}_t\right) \geq \delta_p, \tag{9}$$

where $\mathbb{P}(\cdot)$ denotes probability, $\bigwedge$ represents a conjunction over the indexed constraints ($\bigvee$ represents a disjunction), and $\delta_s, \delta_p \in [0.5, 1]$. The constraint (8) dictates that the state constraints be satisfied at each time step with a probability of at least δ_s, while the constraint (9) dictates that the state constraints be satisfied over *all* time steps with a probability of at least δ_p.

Consider the expected dynamics (1),

$$\widehat{x}_{t+1} = A\widehat{x}_t + Bu_t. \tag{10}$$

Since there is uncertainty in the state, it is assumed sufficient for the expected mean $\widehat{x}_t$ to reach the goal region $\mathcal{X}_{\text{goal}} \subset \mathbb{R}^{n_x}$; denote

$$t_f = \inf\{t \in \mathbb{Z}_{0,\infty} \mid \widehat{x}_t \in \mathcal{X}_{\text{goal}}\}. \tag{11}$$

The path planner seeks to approximately solve the optimal control problem

$$\min_{u_t} \quad \phi_f(\widehat{x}_{t_f}, \mathcal{X}_{\text{goal}}) + \sum_{t=0}^{t_f-1} \phi(\widehat{x}_t, \mathcal{X}_{\text{goal}}, u_t) \tag{12}$$

$$\text{s.t.} \quad (1), (5), (8), (9), (10), (11), \tag{13}$$

where ϕ, ϕ_f are cost functions to be optimized. These functions may be generalized to more complex forms, such as incorporating the state and input constraints; in this work, they are used to penalize risk (Section VI). By using $\widehat{x}_t$ rather than x_t within the objective (12), the stochastic elements of this optimization manifest themselves only in the chance constraints (8)-(9).

IV. Chance Constraints

This section reviews and expands the chance constraint formulation[2] previously developed for the CC-RRT algorithm to consider uncertainty in both vehicle state and obstacle placement.[4] Two expansions to the formulation are presented here. First, it is shown how to guarantee probabilistic

path-wide feasibility via the bound δ_p. Second, the environment boundaries $\mathcal{X}$ are also considered as a probabilistic constraint, rather than a nominal constraint on the conditional mean.[4]

Given a sequence of inputs $u_0, \ldots, u_{t_f-1}$, the distribution of the state x_t, represented as the random variable $\mathbf{X}_t$, can be shown to be Gaussian:[2]

$$\begin{aligned} P(\mathbf{X}_t|u_0,\ldots,u_{t_f-1}) &\sim P(\mathbf{X}_t|u_0,\ldots,u_{t-1}) \\ &\sim \mathcal{N}(\hat{x}_t, P_{x_t}) \ \ \forall\ t \in \mathbb{Z}_{0,N}, \end{aligned} \tag{14}$$

where the mean $\hat{x}_t$ and covariance P_{x_t} can be represented implicitly as

$$\widehat{x}_{t+1} = A\widehat{x}_t + Bu_t \ \ \forall\ t \in \mathbb{Z}_{0,t_f-1}, \tag{15}$$

$$P_{x_{t+1}} = AP_{x_t}A^T + GP_wG^T \ \ \forall\ t \in \mathbb{Z}_{0,t_f-1}. \tag{16}$$

Eq. (15) effectively updates the distribution mean $\widehat{x}_t$ using the disturbance-free dynamics (10), and (16) is independent of the input sequence and thus can be computed *a priori* off-line.

Consider the chance constraints (8)-(9), restated in terms of constraint violation as

$$\mathbb{P}\left(x_t \notin \mathcal{X}_t\right) \leq 1-\delta_s, \ \ \forall\ t \in \mathbb{Z}_{0,t_f}, \tag{17}$$

$$\mathbb{P}\left(\bigvee_{t=0}^{t_f} x_t \notin \mathcal{X}_t\right) \leq 1-\delta_p. \tag{18}$$

To render these constraints tractable, it is desirable to decompose them, first by time step, then by obstacle. Temporal independence *cannot* be assumed in the case of (18), but Boole's bound[2] can be applied to upper-bound its probability:

$$\mathbb{P}\left(\bigvee_{t=0}^{t_f} x_t \notin \mathcal{X}_t\right) \leq \sum_{t=0}^{t_f} \mathbb{P}\left(x_t \notin \mathcal{X}_t\right). \tag{19}$$

It is similarly the case that the probabilities of violating each component of the state constraints (4) are not independent. However, Boole's bound can again be applied, yielding

$$\mathbb{P}\left(x_t \notin \mathcal{X}_t\right) \leq \mathbb{P}\left(x_t \notin \mathcal{X}\right) + \sum_{j=1}^{n_o} \mathbb{P}\left(x_t \in \mathcal{X}_{jt}\right), \ \ \forall\ t \in \mathbb{Z}_{0,t_f}. \tag{20}$$

Consider the jth obstacle at the tth timestep. Because the obstacle is polyhedral, it can be represented through the conjunction of linear inequalities

$$\bigwedge_{i=1}^{n_j} a_{ij}^T(x_t - c_{ijt}) < 0 \ \ \forall\ t \in \mathbb{Z}_{0,t_f}, \tag{21}$$

where n_j is the number of constraints defining the jth obstacle, and c_{ijt} is a point nominally (*i.e.,* $c_{jt} = \widehat{c}_{jt}$) on the ith constraint at time step t; a_{ij} is not dependent on t, since the obstacle shape and orientation are fixed. To avoid the jth obstacle at the tth timestep, the system must satisfy the disjunction

$$\bigvee_{i=1}^{n_j} a_{ij}^T(x_t - c_{ijt}) \geq 0. \tag{22}$$

To avoid the obstacle, it is sufficient to not satisfy any one of the constraints in the conjunction (21); all constraints must be satisfied for a collision. Thus it is true that

$$\begin{aligned}\mathbb{P}(\text{collision with } j\text{th obstacle at } t\text{th time step}) &= \mathbb{P}\left(\bigwedge_{i=1}^{n_j} a_{ij}^T(\mathbf{X}_t - \mathbf{C}_{ijt}) < 0\right) \\ &\leq \mathbb{P}\left(a_{ij}^T(\mathbf{X}_t - \mathbf{C}_{ijt}) < 0\right) \quad \forall\, i \in \mathbb{Z}_{1,n_j},\end{aligned} \tag{23}$$

where $\mathbf{C}_{ijt} = c_{ijt} + (\mathbf{C}_{jt} - \widehat{c}_{jt})$ is a random variable due to (6)-(7).

Suppose it is desired that the probability of collision with the jth obstacle at the tth time step be less than or equal to some quantity Δ. To ensure this from (23), it is only necessary to show that one of the constraints for the obstacle is satisfied with probability less than or equal to Δ:

$$\bigvee_{i=1}^{n_j} \mathbb{P}\left(a_{ij}^T(\mathbf{X}_t - \mathbf{C}_{ijt}) < 0\right) \leq \Delta. \tag{24}$$

Via appropriate change of variables, it has been shown[4] that the constraints (22) are probabilistically satisfied for the true state x_t if the conditional mean $\widehat{x}_t$ satisfies the modified constraints

$$\bigvee_{i=1}^{n_j} a_{ij}^T(\widehat{x}_t - c_{ijt}) \geq \bar{b}_{ijt} \equiv \sqrt{2}P_v \text{erf}^{-1}\left(1 - 2\Delta\right), \tag{25}$$

$$P_v = \sqrt{a_{ij}^T(P_{x_t} + P_{c_{jt}})a_{ij}}. \tag{26}$$

The term $\bar{b}_{ijt}$ represents the amount of *deterministic* constraint tightening necessary to ensure *probabilistic* constraint satisfaction.

Next, consider the polyhedral state constraints $\mathcal{X}$, represented as the conjunction

$$\bigwedge_{i=1}^{n_E} a_{i0}^T(x_t - c_{i0}) < 0 \quad \forall\, t \in \mathbb{Z}_{0,t_f}, \tag{27}$$

where n_E is the number of constraints defining $\mathcal{X}$, and c_{i0} is a point on the ith constraint. Because $\mathcal{X}$ is deterministic and time-invariant, c_{i0} is also deterministic and time-invariant. Violation of the constraints $\mathcal{X}$ at the tth timestep is equivalent to the disjunction of constraints

$$\bigvee_{i=1}^{n_E} a_{i0}^T(x_t - c_{i0}) \geq 0. \tag{28}$$

Boole's bound can be applied one more time, such that

$$\mathbb{P}\left(\bigvee_{i=1}^{n_E} a_{i0}^T(\mathbf{X}_t - c_{i0}) \geq 0\right) \leq \sum_{i=1}^{n_E} \mathbb{P}\left(a_{i0}^T(\mathbf{X}_t - c_{i0}) \geq 0\right). \tag{29}$$

Suppose it is desired that the probability of violating the ith constraint of $\mathcal{X}$ at the tth time step be less than or equal to some quantity Δ_0. By applying a similar change of variable, this condition is met by satisfying the modified constraint

$$a_{i0}^T(\widehat{x}_t - c_{i0}) < -\bar{b}_{i0} \equiv -\sqrt{2}P_v \text{erf}^{-1}\left(1 - 2\Delta_0\right), \tag{30}$$

$$P_v = \sqrt{a_{i0}^T P_{x_t} a_{i0}}. \tag{31}$$

Since P_{x_t} can be computed off-line, the tightened constraints (25),(30) can be computed off-line, implying that the complexity of the nominal formulation need not increase when chance constraints

are incorporated. In a similar manner as in Blackmore *et al.*,[2] probabilistic feasibility of any state or state sequence can be checked via these tightened, deterministic constraints. However, this approach can be heavily conservative, as risk must be pre-allocated to each time step, obstacle, and constraint, with limited knowledge of how much is needed for each.

Alternatively, a more precise bound can be identified online for the probability of collision, both for each time step and the entire trajectory, by computing bounds on the probability of satisfying each individual constraint.[4] This operation is only possible via trajectory-wise constraint checking, as in a sampling-based algorithm such as RRT. In addition to being used to satisfy (17)-(18), these bounds can also be used to penalize risky behavior as a soft constraint, if desired (Section VI). This dynamic assignment of risk to each constraint uses similar logic as iterative risk allocation (IRA);[11] however, whereas IRA iterates on the risk allocation for successive optimizations, a sampling-based algorithm can directly compute an appropriate risk allocation for each constraint.

Consider the ith constraint of the jth obstacle at time step t, as specified in (21). Let $\Delta_{ijt}(\widehat{x}, P_x)$ denote a bound on the probability that this constraint is satisfied for a Gaussian distribution with mean $\widehat{x}$ and covariance P_x; this has been shown to be[4]

$$\Delta_{ijt}(\hat{x}_t, P_{x_t}) = \frac{1}{2}\left(1 - \text{erf}\left[\frac{a_{ij}^T(\hat{x}_t - c_{ijt})}{\sqrt{2a_{ij}^T(P_{x_t} + P_{c_{jt}})a_{ij}}}\right]\right). \tag{32}$$

Through a similar process, let Δ_{i0t} denote the probability that the ith constraint of $\mathcal{X}$ is violated at time step t for a Gaussian distribution with mean $\widehat{x}$ and covariance P_x; then

$$\Delta_{i0t}(\hat{x}_t, P_{x_t}) = \frac{1}{2}\left(1 - \text{erf}\left[\frac{a_{i0}^T(c_{i0} - \hat{x}_t)}{\sqrt{2a_{i0}^T(P_{x_t})a_{i0}}}\right]\right). \tag{33}$$

These components can then be inserted into each usage of Boole's bound (19),(20),(29) to directly bound the probabilities of constraint violation. Define the terms

$$\Delta_{0t}(\hat{x}_t, P_{x_t}) = \sum_{i=1}^{n_E} \Delta_{i0t}(\hat{x}_t, P_{x_t}), \tag{34}$$

$$\Delta_{jt}(\hat{x}_t, P_{x_t}) = \min_{i=1,\ldots,n_j} \Delta_{ijt}(\hat{x}_t, P_{x_t}), \tag{35}$$

$$\Delta_t(\hat{x}_t, P_{x_t}) = \Delta_{0t}(\hat{x}_t, P_{x_t}) + \sum_{j=1}^{n_o} \Delta_{jt}(\hat{x}_t, P_{x_t}), \tag{36}$$

$$\Delta(\hat{x}_t, P_{x_t}) = \sum_{t=0}^{t_f} \Delta_t(\hat{x}_t, P_{x_t}). \tag{37}$$

Then (19) can be represented as

$$\begin{aligned}
\mathbb{P}\left(\bigvee_{t=0}^{t_f} x_t \notin \mathcal{X}_t\right) &\leq \sum_{t=0}^{t_f}\left[\mathbb{P}\left(x_t \notin \mathcal{X}\right) + \sum_{j=1}^{n_o}\mathbb{P}\left(x_t \in \mathcal{X}_{jt}\right)\right] \\
&\leq \sum_{t=0}^{t_f}\left[\sum_{i=1}^{n_E}\mathbb{P}\left(a_{i0}^T(\mathbf{X}_t - c_{i0}) \geq 0\right) + \sum_{j=1}^{n_o}\min_{i=1,\ldots,n_j}\mathbb{P}\left(a_{ij}^T(\mathbf{X}_t - \mathbf{C}_{ijt}) < 0\right)\right] \\
&= \sum_{t=0}^{t_f}\left[\sum_{i=1}^{n_E}\Delta_{i0t}(\widehat{x}_t, P_{x_t}) + \sum_{j=1}^{n_o}\min_{i=1,\ldots,n_j}\Delta_{ijt}(\widehat{x}_t, P_{x_t})\right] \\
&= \sum_{t=0}^{t_f}\left[\Delta_{0t}(\widehat{x}_t, P_{x_t}) + \sum_{j=1}^{n_o}\Delta_{jt}(\widehat{x}_t, P_{x_t})\right] = \sum_{t=0}^{t_f}\Delta_t(\widehat{x}_t, P_{x_t}) = \Delta(\widehat{x}_t, P_{x_t}). \quad (38)
\end{aligned}$$

Thus, to satisfy the path-wise chance constraint (9), it is sufficient to show that

$$\Delta(\widehat{x}_t, P_{x_t}) \leq 1 - \delta_p. \quad (39)$$

By using a subset of the derivation above, it can be similarly shown that

$$\Delta_t(\widehat{x}_t, P_{x_t}) \leq 1 - \delta_s, \quad \forall\ t \in \mathbb{Z}_{0,t_f} \quad (40)$$

is a sufficient condition for satisfying the time-step-wise chance constraints (8).

Both sets of risk bounds are conservative approximations of the true risk of constraint violation; the degree of conservatism increases in both the number of obstacles (for both path-wise and time-step-wise probabilistic feasibility) and the number of time steps (for path-wise probabilistic feasibility). As such, (40) typically provides a less conservative estimate of the true risk environment than (39), though either can be used within the CC-RRT and CC-RRT* algorithms.

V. CC-RRT* Algorithm

This section introduces the CC-RRT* algorithm, a real-time algorithm designed to quickly identify and refine probabilistically feasible trajectories in the presence of time-varying and/or uncertain constraints. Through tight integration of sampling-based planning, chance constraints, and the asymptotic optimality of the RRT* framework,[7] this algorithm achieves both probabilistic feasibility and asymptotic optimality while maintaining real-time tractability.

This algorithm builds upon the chance constrained RRT (CC-RRT) algorithm, which enables the use of probabilistic constraints.[4] Whereas the traditional RRT algorithm grows a tree of states which are known to be feasible, CC-RRT grows a tree of state distributions known to satisfy an upper bound on probability of collision. Additionally, the CC-RRT* algorithm has been designed to fit into the constraints of the RRT* framework,[7,8] such that guarantees on asymptotic optimality are maintained. The RRT* framework expands on RRT by "rewiring" connections within the tree in order to optimize a cost function satisfying certain criteria (Section VI). The CC-RRT* algorithm expands this framework by checking *probabilistic* feasibility, via the chance constraints (39) and/or (40).

To grow a tree of dynamically feasible trajectories, it is necessary for the CC-RRT* algorithm to have an accurate model of the vehicle dynamics (1) for simulation. For each simulated trajectory, the CC-RRT* algorithm propagates the predicted state distribution, which under the assumption of Gaussian uncertainty is itself Gaussian. Thus, at each time step of each simulated trajectory, it is

only necessary to propagate the state conditional mean (15) and covariance (16)[a]. In this manner, the distribution mean $\widehat{x}_t$ replaces the role of the true state x_t in the nominal RRT algorithm.

The CC-RRT* tree is denoted by $\mathcal{T}$, consisting of $|\mathcal{T}|$ nodes. Each node N of the tree $\mathcal{T}$ consists of a sequence of state distributions, characterized by a distribution mean $\widehat{x}$ and covariance P. A sequence of means and covariances is denoted by $\bar{\sigma}$ and $\bar{\Pi}$, respectively. The final mean and covariance of a node's sequence are denoted by $\widehat{x}[N]$ and $P[N]$, respectively.

The cost function (12), with $\phi_f = 0$, is utilized in two forms within the CC-RRT* algorithm. Let $t[N]$ denote the terminal time step for node N. The notation

$$J[N] = dt \sum_{t=0}^{t[N]} f(\widehat{x}_t, P_t), \tag{41}$$

denotes the entire path cost from the starting state to the terminal state of node N, where dt is the time step duration and f is the per-timestep cost objective specified by the user (Section VI). For the state distribution sequence $(\bar{\sigma}, \bar{\Pi})$, the notation

$$\Delta J(\bar{\sigma}, \bar{\Pi}) = dt \sum_{(\widehat{x},P)\in(\bar{\sigma},\bar{\Pi})} f(\widehat{x}, P) \tag{42}$$

denotes the cost of that sequence. Eq. (41) can be constructed recursively by utilizing (42): if $(\bar{\sigma}, \bar{\Pi})$ denotes the trajectory of node N with parent N_{parent}, then

$$J[N] = J[N_{\text{parent}}] + \Delta J(\bar{\sigma}, \bar{\Pi}). \tag{43}$$

The CC-RRT* algorithm consists of two primary components, enabling its use in real-time operations.[6] The first component, the tree expansion step, is used to incrementally grow the tree by simulating new trajectories; it is generally run continuously, using any available computational resources. The second component, the execution loop, periodically selects the best available path for execution, in addition to updating the current state of the vehicle, environment, and tree. In this manner, portions of the tree which remain dynamically feasible are retained for future growth cycles.

The tree expansion step for CC-RRT* is given by Algorithm 1. It starts with the current tree $\mathcal{T}$ at time step t (line 1 of Algorithm 1) and seeks to add additional nodes to the tree, possibly removing some in the process via rewiring. First, a state is sampled from the environment via the "Sample" function (line 2). The function $x = \text{Sample}()$ must return independent and identically distributed samples from X_{free}, the obstacle-free portion of the state space.[8] It is assumed in the problem statement (Section III) that all state elements are bounded. Thus, the approach utilized here is to sample each state element independently, with each realization having a non-zero probability, then filter out any infeasible samples.

Next, a node in $\mathcal{T}$ nearest to x_{samp} in terms of some distance metric is identified via the "Nearest" function (line 3). The function $N = \text{Nearest}(\mathcal{T}, x)$ uses the Euclidean norm metric via[7]

$$N = \text{Nearest}(\mathcal{T}, x) = \arg\min_{N\in\mathcal{T}} \|x - \widehat{x}[N]\|. \tag{44}$$

The Steering law "Steer" is then applied to steer the terminal state mean $\widehat{x}[N_{\text{nearest}}]$ to x_{samp} (line 4). The function $(\bar{\sigma}, \bar{\Pi}) = \text{Steer}(x, P, y)$ returns a sequence of state distributions, characterized by their means $(\bar{\sigma})$ and covariances $(\bar{\Pi})$, which originates at state x and terminates at state y. The state distributions must be dynamically feasible: the mean sequence $\bar{\sigma}$ must satisfy (15) with initial

[a]In subsequent work, covariances of the form P_{x_t} will often be rewritten as P_t to simplify presentation.

Algorithm 1 CC-RRT*, Tree Expansion

1: Inputs: tree $\mathcal{T}$, current time step t
2: $x_{\text{samp}} \leftarrow \text{Sample}()$
3: $N_{\text{nearest}} \leftarrow \text{Nearest}(\mathcal{T}, x_{\text{samp}})$
4: $(\bar{\sigma}, \bar{\Pi}) \leftarrow \text{Steer}(\widehat{x}[N_{\text{nearest}}], P[N_{\text{nearest}}], x_{\text{samp}})$
5: **if** $\text{ProbFeas}(\bar{\sigma}, \bar{\Pi})$ **then**
6: Create node $N_{\min}\{\bar{\sigma}, \bar{\Pi}\}$
7: $\mathcal{N}_{\text{near}} \leftarrow \text{Near}(\mathcal{T}, x_{\text{samp}}, |\mathcal{T}|)$
8: **for** $N_{\text{near}} \in \mathcal{N}_{\text{near}} \backslash N_{\text{nearest}}$ **do**
9: $(\bar{\sigma}, \bar{\Pi}) \leftarrow \text{Steer}(\widehat{x}[N_{\text{near}}], P[N_{\text{near}}], x_{\text{samp}})$
10: **if** $\text{ProbFeas}(\bar{\sigma}, \bar{\Pi})$ and $J[N_{\text{near}}] + \Delta J(\bar{\sigma}, \bar{\Pi}) < J[N_{\min}]$ **then**
11: Replace $N_{\min}$ with new node $N_{\min}\{\bar{\sigma}, \bar{\Pi}\}$
12: **end if**
13: **end for**
14: Add $N_{\min}$ to $\mathcal{T}$
15: **for** $N_{\text{near}} \in \mathcal{N}_{\text{near}} \backslash \text{Ancestors}(N_{\min})$ **do**
16: $(\bar{\sigma}, \bar{\Pi}) \leftarrow \text{Steer}(\widehat{x}[N_{\min}], P[N_{\min}], \widehat{x}[N_{\text{near}}])$
17: **if** $\text{ProbFeas}(\bar{\sigma}, \bar{\Pi})$ and $J[N_{\min}] + \Delta J(\bar{\sigma}, \bar{\Pi}) < J[N_{\text{near}}]$ **then**
18: Delete N_{near} from $\mathcal{T}$
19: Add new node $N_{\text{new}}\{\bar{\sigma}, \bar{\Pi}\}$ to $\mathcal{T}$
20: Update descendants of N_{new} as needed
21: **end if**
22: **end for**
23: **end if**

mean x, while the covariance sequence $\bar{\Pi}$ must satisfy (16) with initial covariance P. Additionally, the inputs u_t applied by this steering law must satisfy the input constraints (5). The choice of steering law is heavily dependent on the dynamics being considered (Section VII).

The resulting distribution sequence is then checked for probabilistic feasibility via the function "ProbFeas" (line 5). The Boolean function $\text{ProbFeas}(\bar{\sigma}, \bar{\Pi})$ returns `true` if the distribution sequence $(\bar{\sigma}, \bar{\Pi})$ satisfies the probabilistic feasibility conditions (39)-(40), and `false` otherwise; its details are given in Algorithm 2. The subroutine iterates over all elements of $(\bar{\sigma}, \bar{\Pi})$ (line 2 of Algorithm 2); for the kth element with mean $\widehat{x}$ and covariance P (line 3), $\Delta_t(\widehat{x}, P)$ and/or $\Delta(\widehat{x}, P)$ (depending on which chance constraints are being enforced) are computed (line 4). It is assumed that the values of Δ_t at previous time steps are available for the computation of $\Delta(\widehat{x}, P)$. If either value exceeds the maximum allowable risk (line 5), the subroutine returns `false` (line 6); if this does not occur for any state distribution, the subroutine returns `true` (line 9).

If $(\bar{\sigma}, \bar{\Pi})$ is probabilistically feasible, a new node with that distribution sequence is created (line 6 of Algorithm 1), but not yet added to $\mathcal{T}$. Instead, nearby nodes are identified for possible connections via the "Near" function $\mathcal{N} = \text{Near}(\mathcal{T}, x, n)$ (line 7), which returns a subset of nodes $\mathcal{N} \subseteq \mathcal{T}$. To enable probabilistic asymptotic optimality guarantees, CC-RRT* uses the implementation[8]

$$\mathcal{N} = \text{Near}(\mathcal{T}, x, n) \equiv \{N \in \mathcal{T} \mid \|\widehat{x}[N] - x\| \leq r_n\}, \tag{45}$$

$$r_n = \min\left\{\left(\frac{\gamma}{\zeta_d}\frac{\log n}{n}\right)^{1/d}, \ \mu\right\}, \tag{46}$$

where $\mu > 0$ is a maximum radius specified by the user, and ζ_d denotes the volume of a unit ball in $\mathbb{R}^d$. The parameter $\gamma > 0$ is chosen such that[8]

$$\gamma \geq 2^d\left(1 + \frac{1}{d}\right)\mu(\mathcal{X}_{\text{free}}), \tag{47}$$

where $\mu(V)$ denotes the volume of V and $\mathcal{X}_{\text{free}}$ denotes the obstacle-free configuration space.

Algorithm 2 ProbFeas

1: Inputs: dynamically feasible K-time-step sequence of means $\bar{\sigma}$, covariances $\bar{\Pi}$
2: **for** $k = 1$ to K **do**
3: $(\widehat{x}, P) \leftarrow k$th element of $(\bar{\sigma}, \bar{\Pi})$
4: Compute $\Delta_t(\widehat{x}, P)$ using (36) and $\Delta(\widehat{x}, P)$ using (37)
5: **if** $\Delta(\widehat{x}, P) > 1 - \delta_p$ **or** $\Delta_t(\widehat{x}, P) > 1 - \delta_s$ **then**
6: **return** `false`
7: **end if**
8: **end for**
9: **return** `true`

Algorithm 3 CC-RRT*, Execution Loop

1: Initialize tree $\mathcal{T}$ with node $(\hat{x}_0, P_{x_0})$ for $t = 0$
2: **while** $\hat{x}_t \notin \mathcal{X}_{\text{goal}}$ **do**
3: Use observations, if any, to update current state distribution $(\widehat{x}_t, P_t)$ and node N_{root} and/or environment
4: If needed, advance N_{root}, remove infeasible parts of $\mathcal{T}$
5: **while** time remaining for this time step **do**
6: Expand the tree by adding nodes (Algorithm 1)
7: **end while**
8: Identify path $\{N_{\text{root}}, \ldots, N_{\text{target}}\}$ that minimizes (41)
9: **if** no paths exist **then**
10: Apply safety action and **goto** line 18
11: **end if**
12: **for** each node $N\{\bar{\sigma}, \bar{\Pi}\}$ in path **do**
13: **if** ProbFeas$(\bar{\sigma}, \bar{\Pi})$ `false` **then**
14: Remove infeasible portion of path and **goto** line 8
15: **end if**
16: **end for**
17: Execute path
18: $t \leftarrow t + \Delta t$
19: **end while**

Once the nearby nodes $\mathcal{N}$ are identified, CC-RRT* seeks to identify the lowest-cost, probabilistically feasible connection from those nodes to x_{samp} (lines 8–13). For each possible connection, a distribution sequence is simulated via the steering law (line 9). If the resulting sequence is probabilistically feasible, and the cost of that node – represented as the sum $J[N_{\text{near}}] + \Delta J(\bar{\sigma}, \bar{\Pi})$, via (43) – is lower than the cost of N_{min} (line 10), then a new node with this sequence replaces N_{min} (line 11). The lowest-cost node is ultimately added to $\mathcal{T}$ (line 14).

Finally, a rewiring operation is performed based on attempting connections from the new node N_{min} to nearby nodes (lines 15–22), ancestors excluded (line 15). A distribution sequence is sampled via the steering law from N_{min} to the terminal state of each nearby node N_{near} (line 16). If the resulting sequence is probabilistically feasible, and the cost of that node is lower than the cost of N_{near}, then a new node with this distribution sequence replaces N_{near} (lines 18–19).

By using the RRT* rewiring mechanism with an exact steering law, all descendants of a rewired node remain dynamically feasible. As in that algorithm, the reduced path cost at the rewired node N_{new} should be propagated downward through all descendants (line 20). On the other hand, the terminal covariance/risk may change due to rewiring, implying that it should also be propagated to – and probabilistic feasibility re-checked at – descendant nodes. Because all uncertainty covariances can be computed and time-indexed off-line (Section IV), these re-checks can be computed efficiently. Additionally, under some conditions (*e.g.,* non-increased uncertainty after rewiring), feasibility re-checks are often not necessary to ensure probabilistic feasibility of descendants (Section VII). This will be explored further in future work.

The CC-RRT* algorithm's execution loop, which is performed at time intervals of Δt, is given by Algorithm 3. During each cycle, the objective of this algorithm is to identify the lowest-cost path in the tree that is still feasible, and use the remaining time to grow the tree.

If new observations are available for the vehicle's current state and/or environment (such as movement of dynamic obstacles), these may be applied to the tree first, resulting in the update of the current root node N_{root} (line 3). If the terminal covariance $P[N_{\text{root}}]$ changes, it should be propagated through the rest of the tree $\mathcal{T}$. Every time the system advances past a node to one of its children, all other children of that node are no longer dynamically feasible and should be removed

(line 4). For the duration of the time step, the tree is repeatedly expanded using Algorithm 1 (lines 5–7). Following this tree growth, the objective (41) is used to identify the lowest-cost path in the tree (line 8). In practice, only paths which terminate in $\mathcal{X}_{\text{goal}}$ are considered; if no such path exists, the path which terminates closest to the goal region (in terms of Euclidean distance) is selected.

Once a path is chosen, it is re-checked for probabilistic feasibility[6] against the current constraints (lines 12–16). If this path is still probabilistically feasible, it is chosen as the current path to execute (lines 17). Otherwise, the portion of the path that is no longer probabilistically feasible is removed (lines 13–14), and the process is repeated until either a probabilistically feasible path is found or the entire tree is pruned. If the latter case occurs, some "safety" motion primitive (*e.g.,* come to a stop) is applied to attempt to keep the vehicle in a safe state (lines 9–10).

VI. Analysis

This section analyzes the properties of the CC-RRT* algorithm, as presented in Section V. First, it is shown that paths selected by the CC-RRT* algorithm satisfy all constraints for the path planning problem (12)-(13) proposed in Section III, including the chance constraints (8)-(9). A novel cost function is then introduced which incorporates the risk of constraint violation into the objective, acting as a form of soft constraint on the objective. It is shown that this cost function is admissible within the RRT* framework, meaning that the CC-RRT* algorithm is able to minimize the proposed function with asymptotic optimality.

Theorem 1. *All paths selected by Algorithm 3 satisfy the constraints (1),(5),(8),(9).*

Proof. All trajectory segments are generated using the Steer function, which is required to satisfy both (15), which is equivalent to (1) with $w_t = 0$, and (5). For any node $N\{\bar{\sigma}, \bar{\Pi}\}$ to be added to the tree, ProbFeas$(\bar{\sigma}, \bar{\Pi})$ must return `true`. For that to be the case, every state distribution $(\widehat{x}, P) \in (\bar{\sigma}, \bar{\Pi})$ must satisfy $\Delta(\widehat{x}, P) \leq 1 - \delta_p$ and $\Delta_t(\widehat{x}, P) \leq 1 - \delta_s$, *i.e.,* (39)-(40). These conditions remain satisfied after rewiring. From (34)-(38), these are sufficient to satisfy (8),(9). ■

The cost function used is a novel feature of this work, as it explicitly incorporates the risk of constraint violation inherent to chance constraints. It takes the form

$$f(\widehat{x}_t, P_t) = C_T + C_R \Delta_t(\widehat{x}_t, P_t) + C_M \max_{i=\{0,\dots,t\}} \Delta_t(\widehat{x}_t, P_t), \tag{48}$$

where $C_T \geq 0$, $C_R \geq 0$, $C_M \geq 0$, $C_T C_R C_M > 0$. The cost component with coefficient C_T seeks to minimize path duration. The cost component with coefficient C_R represents the accumulated risk across all time steps, as measured by the risk bound $\Delta_t(\widehat{x}_t, P_t)$. Finally, the cost component with coefficient C_M penalizes the maximum risk bound encountered at any time step along the path.

Theorem 2. *The cost function (41), using (48), is an admissible cost function for RRT*.*

Overview of proof. Using the framework established by Karaman and Frazzoli,[8] there are three conditions the cost function (41), using (48), must satisfy to be admissible.

First, the cost must be monotonic: if $\bar{\sigma}_1|\bar{\sigma}_2$ denotes the concatenation of path segments $\bar{\sigma}_1$ and $\bar{\sigma}_2$, then $\Delta J(\bar{\sigma}_1) \leq \Delta J(\bar{\sigma}_1|\bar{\sigma}_2)$. Since $C_T, C_R, C_M \geq 0$ and $\Delta_t(\widehat{x}_t, P_t) \geq 0$, then $f(\widehat{x}_t, P_t) \geq 0 \; \forall \; (\widehat{x}_t, P_t)$, implying monotonicity.

Second, the cost must be additive, *i.e.,* $c(\bar{\sigma}_1|\bar{\sigma}_2) = c(\bar{\sigma}_1) + c(\bar{\sigma}_2)$. This is verified via (43). Note that finding a maximum-risk state further down a path does not retroactively increase the value of $\max_{i=\{0,\dots,t\}} (\Delta_t(\widehat{x}_t, P_t))$ at previous times.

Finally, the cost must be Lipschitz continuous: there exists some $\kappa > 0$ such that

$$|\Delta J(\bar{\sigma}_1) - \Delta J(\bar{\sigma}_2)| \leq \kappa \sup_{\tau \in [0,1]} \|\widehat{x}_1(\tau) - \widehat{x}_2(\tau)\|, \tag{49}$$

where $\tau \in [0,1]$ parametrizes the state distribution trajectories $\widehat{x}_1(\cdot)$ and $\widehat{x}_2(\cdot)$ for $\bar{\sigma}_1$ and $\bar{\sigma}_2$, respectively. The C_T-component of the cost function is clearly Lipschitz continuous. For the C_R-component, consider $\Delta_t(\widehat{x}_t, P_t)$, which is a sum of terms (34) and (35) defined by (32) and (33), respectively. The error function $\text{erf}(\cdot)$ is continuous with a bounded slope, and both a_{ij} and a_{i0} are also bounded. Thus smooth shifts in the state uncertainty distribution via $\widehat{x}_t$ and/or P_{x_t} yield smooth variations on $\Delta_{ijt}(\widehat{x}_t, P_{x_t})$ and $\Delta_{i0t}(\widehat{x}_t, P_{x_t})$, implying that the C_R-component is also Lipschitz continuous. Finally, the C_M-component applies the maximum operator on the C_R-component, which is Lipschitz continuous, and thus is Lipschitz continuous as well. Thus, the sum of these terms, which compromises the cost function, is Lipschitz continuous.

As the above conditions are satisfied by the cost function (41), (48), it is thus admissible for RRT*. ■

VII. Simulation Results

This section presents simulation results which demonstrate the effectiveness of the CC-RRT* algorithm in efficiently identifying smooth, robust trajectories subject to both internal and external uncertainties. Applying CC-RRT* with a time-based objective (*e.g.,* $C_R, C_M = 0$) is shown to generate trees of trajectories satisfying all robustness chance constraints, yielding an asymptotically optimal trajectory that is both probabilistically and dynamically feasible. As the likelihood of constraint violation tends to increase with proximity to obstacles, the chance constraints will typically be active in the final trajectory, with solutions often approaching the maximum allowable risk. Alternatively, by incorporating measures of risk within the objective (*e.g.,* $C_R, C_M > 0$), the resulting trees and trajectories demonstrate more risk-averse behavior, avoiding riskier actions unless deemed necessary to reduce path duration, as determined by the relative coefficient weights.

Six variants of the RRT algorithm are compared throughout this section: RRT,[3] RRT*,[8] CC-RRT,[4] CC-RRT-Risk (same as CC-RRT, but utilizing (48) with $C_T, C_R, C_M > 0$), CC-RRT* (Algorithm 1 with $C_T > 0$, $C_R = C_M = 0$), and CC-RRT*-Risk ($C_T, C_R, C_M > 0$).

A. Illustrative Scenario

Consider the 2D single integrator dynamics

$$x_{t+1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} dt & 0 \\ 0 & dt \end{bmatrix} u_t + w_t,$$

where $dt = 0.1$ s. The position variables $x_t = (p_t^x, p_t^y)$ are constrained within a bounded, two-dimensional 10m $\times$ 10m environment, containing four obstacles with uncertain placement (Figure 1). The velocity inputs $u_t = (v_t^x, v_t^y)$ are subject to the input constraints $\mathcal{U} = \{(v_t^x, v_t^y) \mid |v_t^x| \leq \bar{v}, |v_t^y| \leq \bar{v}\}$, where $\bar{v} = 0.5$ m/s. The system is modeled as a circular object, considered as a point mass during planning by expanding all obstacles by its radius.

The system is subject to three forms of uncertainty. First, the initial state x_0 is subject to localization error (3) more prominent in the y-direction,

$$x_0 \in \mathcal{N}(\widehat{x}_0, P_{x_0}^{(1)}), \quad P_{x_0}^{(1)} = 10^{-5} \begin{bmatrix} 5 & 0 \\ 0 & 30 \end{bmatrix}.$$

At each time step, the system is subject to process noise (2) more prominent in the x-direction,

$$w_t \in \mathcal{N}(0, P_w^{(1)}), \quad P_w^{(1)} = 10^{-5} \begin{bmatrix} 30 & 0 \\ 0 & 5 \end{bmatrix}.$$

Finally, the placement of each obstacle is itself uncertain. At all time steps, the displacement of the jth obstacle is governed by (7), where

$$c_{jt} \in \mathcal{N}(\widehat{c}_{jt}, P_{c_j}^{(1)}), \quad P_{c_j}^{(1)} = \begin{bmatrix} \sigma_j & 0 \\ 0 & \sigma_j \end{bmatrix}, \quad \forall\, t,$$

where $\sigma_j > 0$. In this environment (Figure 1 – obstacles are placed at their means $\widehat{c}_{jt}$), $\sigma_j = 0.2$ for the upper-left obstacle, $\sigma_j = 0.1$ for the bottom-right obstacle, and $\sigma_j = 0.001$ for the other two obstacles. The system is required to satisfy a minimum probability of constraint violation at each time step of $\delta_s = 0.9$; no path-wise probability bound is imposed.

In the following scenarios, the external uncertainty is time-invariant, while the internal uncertainty is monotonically increasing (without bound) and path-independent. Thus any nodes re-wired to a shorter path (*i.e.,* via the CC-RRT* objective) are guaranteed to be less uncertain, and probabilistic feasibility of all descendant nodes is assured without re-checking. Even if a node is rewired to a longer path via the risk-based objective of CC-RRT*-Risk, the implied risk reduction in the rewired path means descendant nodes are likely to demonstrate more robust feasibility.

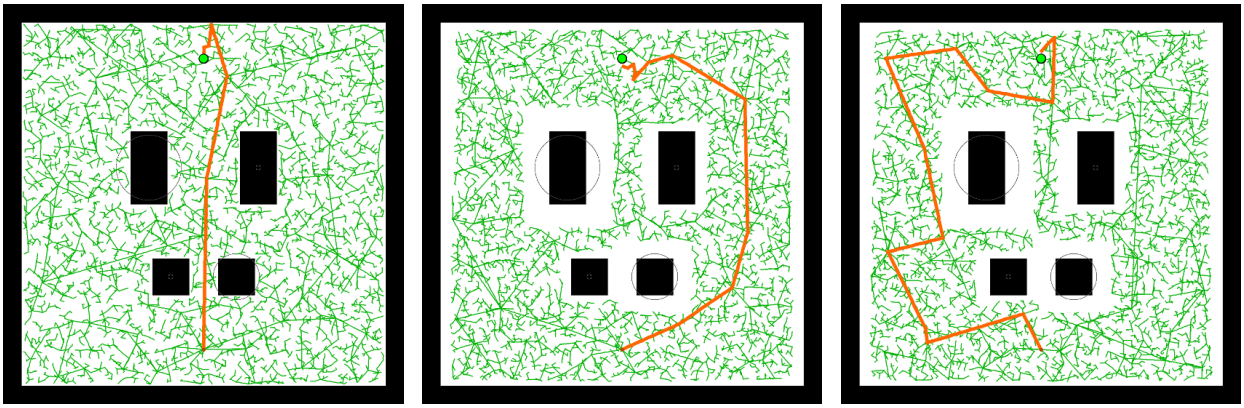
The 2D position is sampled uniformly within the bounds of the feasible 2D environment. The steering law simply draws a line connecting the old and new positions; the system traverses this line at speed $\bar{v}$. A path is considered to reach the goal is the final position is within 0.25m of the goal location. Finally, the nearby node function uses a maximum radius $\mu = 1$ m, while $\mu(\mathcal{X}_{\text{free}}) = A_{\text{free}}$, where A_{free} denotes the 2D area of the environment.

Figure 1 shows typical trees and solution paths returned by each algorithm after 5000 nodes of tree growth, via Algorithm 1. As expected, the RRT-based algorithms (Figures 1(a), 1(b), and 1(c)) generate trees which are relatively random and unorganized in their path structure, yielding non-smooth paths and short path segments. In contrast, the RRT*-based algorithms (Figures 1(d), 1(e), and 1(f)) attempt to rewire the tree to reduce path costs every time a new sample is added, yielding more organized trees with longer node path segments. The cost-minimizing paths identified by these algorithms thus tend to be relatively smooth, as demonstrated in the images.

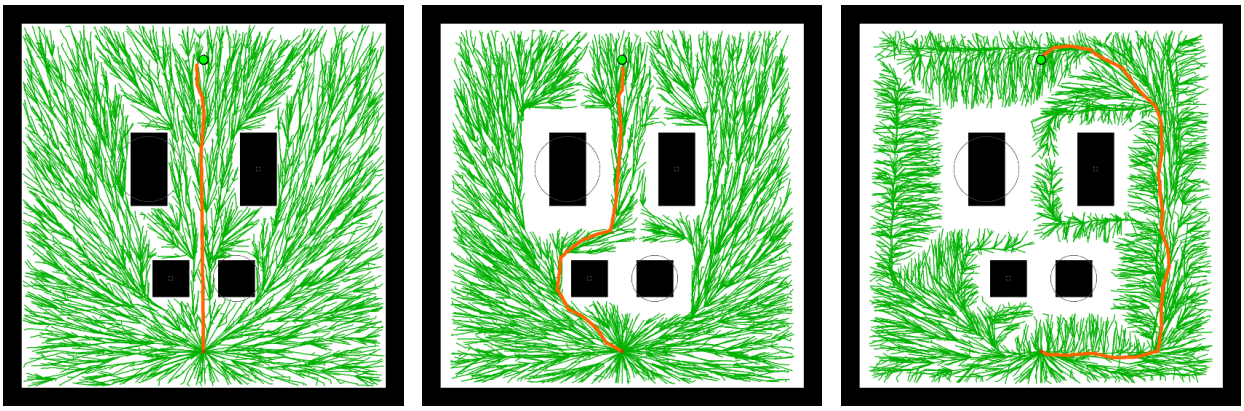
Both RRT (Figure 1(a)) and RRT* (Figure 1(d)) identify short paths to the goal which take the system between all obstacles. However, both algorithms do not consider the risk of constraint violation, and thus are likely to select risky behaviors in order to minimize path duration. In particular, the uncertainty in the placement of the bottom-right obstacle presents a high chance of collision for the system as it passes nearby. The other algorithms only add trajectories to the tree if they satisfy (40) via Algorithm 2. A "buffer" containing no (probabilistically feasible) tree paths can be seen around each obstacle and the environment boundaries for those algorithms, reflecting regions where the cumulative uncertainty violates the maximum-risk bound. This buffer is larger for the more uncertain obstacles, and increases with distance from the starting position as process noise accumulates. No probabilistically feasible path exists that takes the system between the two lower obstacles, implying the paths chosen by RRT and RRT* violate probabilistic feasibility.

CC-RRT* (Figure 1(e)) identifies a minimum-cost solution path which takes the system around both lower obstacles to the left, to avoid the uncertain bottom-right obstacle, then passes between the upper obstacles at a sufficient distance to remain probabilistically feasible. Such a path would also be feasible for CC-RRT (Figure 1(b)), but is unlikely to be sampled. Because CC-RRT is not an asymptotically optimal algorithm, it cannot refine tree paths once generated; thus solution paths are highly dependent on the random placement of initial tree samples. For example, in Figure 1(b)), the minimum-time path to goal passes around all obstacles to the right, even though a path between them would have been probabilistically feasible if identified.

Both CC-RRT* and CC-RRT*-Risk are subject to the same probabilistic constraints; however, the shapes of both the resulting trees and solution paths are significantly affected by the use

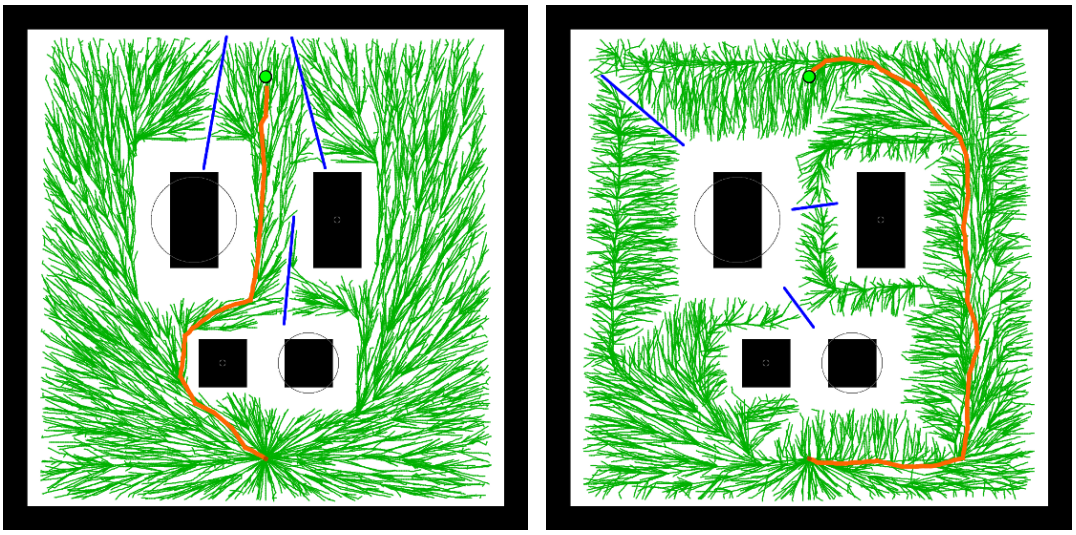


(a) RRT (b) CC-RRT (c) CC-RRT-Risk



(d) RRT* (e) CC-RRT* (f) CC-RRT*-Risk

Figure 1. Demonstrative 5000-node trees generated by each algorithm for the single integrator. The objective is to plan a path from the start (brown dot, bottom) to the goal (lime green circle, top); the minimum-cost path after 5000 nodes is shown in orange. The 2-σ uncertainty ellipse is shown for the placement uncertainty of each obstacle.



(a) CC-RRT* (b) CC-RRT*-Risk

Figure 2. Figures 1(e) and 1(f), with homotopic boundaries marked in blue.

of different cost objectives. Whereas CC-RRT* seeks the geometrically-shortest path subject to the robustness constraints (Figure 1(e)), CC-RRT*-Risk instead identifies a path which passes around all obstacles to the right at a significant distance (Figure 1(f)). While the resulting path is similar to the example path generated for CC-RRT (Figure 1(b)), it is generated via the rewiring process, which yields asymptotically optimal convergence. Repeated executions of the algorithm for this scenario would yield very similar final paths (Section VII-B). (Using the risk-based objective function has little effect on CC-RRT, as observed by CC-RRT-Risk in Figure 1(c).)

Because the trees for CC-RRT* and CC-RRT*-Risk have been rewired based on different cost functions, the resulting trees show significant qualitative differences. Paths passing around obstacles in the CC-RRT* tree (Figure 1(e)) tend to travel parallel to the obstacle surfaces, in order to minimize path duration. On the other hand, paths passing around obstacles in the CC-RRT*-Risk tree (Figure 1(f)) tend to travel *perpendicular* to the obstacle surfaces, in order to minimize the time spent by trajectories in higher-risk (and thus higher-cost) regions.

Of particular note are the "homotopic boundaries" of each tree: those boundaries separating portions of the tree that pass around obstacles on differing sides, thus belonging to different homotopies. Figure 2 reproduces Figures 1(e) and 1(f) with homotopic boundaries approximately marked. The boundaries of the CC-RRT* tree (Figure 2(a)) largely follow the Voronoi minimum-distance boundaries, with a slight preference for passing around the lower obstacles on the left. For CC-RRT*-Risk, these boundaries have shifted significantly; for example, along the top of the environment (*i.e.,* above all obstacles), nearly all tree paths approach from the right-hand side.

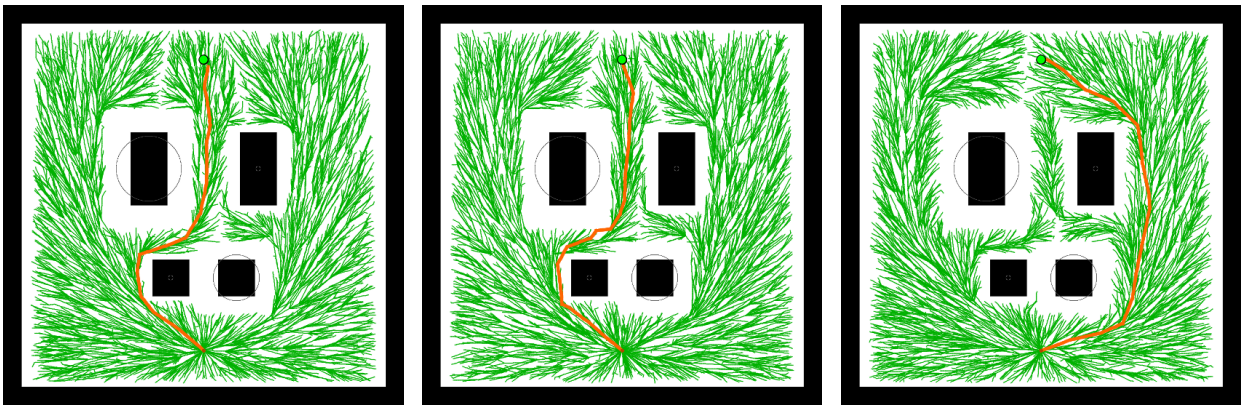
Governing these changes is the trade-off between minimizing path length and minimizing risk, as dictated by the cost coefficients of (48). Consider the ratio of cost coefficients $\gamma = C_R/C_T$, where it is assumed that $C_M = C_R$; Figure 3 shows how the resulting CC-RRT*-Risk trees evolve as γ is varied. For low values of γ (Figure 3(a)), the resulting tree behavior is essentially the same as CC-RRT* ($\gamma = 0$). As γ is increased to 1 (Figure 3(b)), only subtle changes can be observed in the tree and path. When γ is increased to 10 (Figure 3(c)), the upper-right homotopic boundary sweeps past the goal, causing the solution path to shift toward passing all obstacles on the outside. The solution path increases its distance from obstacles as γ increases further (Figures 3(d), 3(e)).

Regardless, it is essential that $C_T > 0$: including a cost term which minimizes path duration acts as a regularization parameter for CC-RRT*, especially in low-risk regions. Figure 3(f) shows the trees and paths that typically result when $C_T = 0$, *i.e.,* $\gamma \to \infty$. In regions where the risk bounds approach zero, tree behavior essentially reverts to the unorganized nature of RRT, resulting in risk-averse paths that are often significantly less smooth.

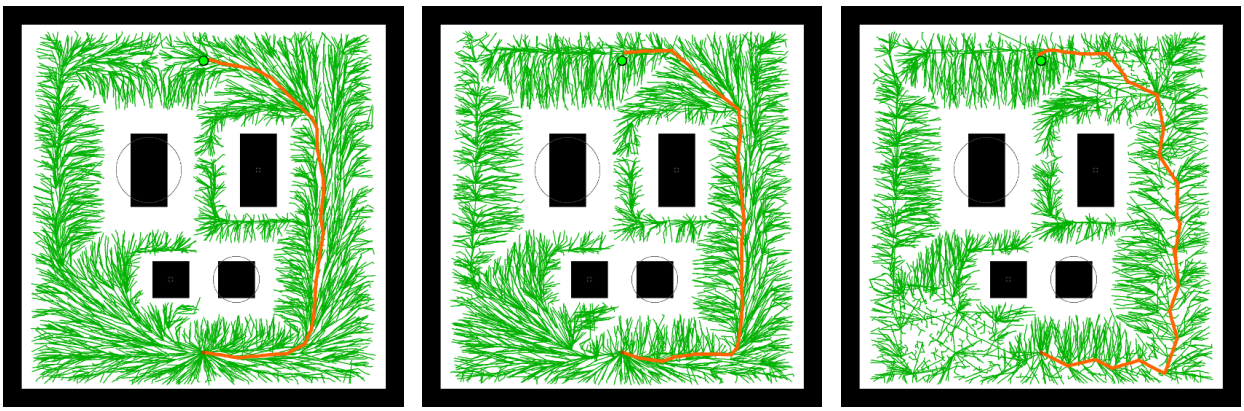
Figure 4(a) depicts a 500-node tree with the $2-\sigma$ uncertainty ellipses visible for each tree node, showing the accumulation of uncertainty over time. This effect is even more pronounced in Figure 4(b), in which the covariances of the localization uncertainty $P_{x_0}^{(1)}$ and model uncertainty $P_w^{(1)}$ have each been scaled up by a factor of 10. The distance the system must maintain from the room boundaries clearly increases as paths move toward the goal. Figure 4(c) adds a path-wise probabilistic feasibility bound of $\delta_p = 0.9$, in addition to a reduced time-step-wise bound $\delta_s = 0.5$. Due to the additional chance constraint (9), the space of feasible solutions here is significantly reduced compared to $\delta_s = 0.9$ (Figure 1(f)), as expected. In all these cases, even as the size and shape of the tree varies considerably, the final path selected is qualitatively unchanged. This final path invariance is due to the risk-based cost objective encouraging risk-averse behavior, even as the hard probabilistic feasibility constraints are varied.

B. Simulation Trials

Consider the same single integrator dynamics of Section VII-A now applied to a different environment containing four obstacles (Figure 6). The environment is 37 feet (11.3 m) long and 18 feet

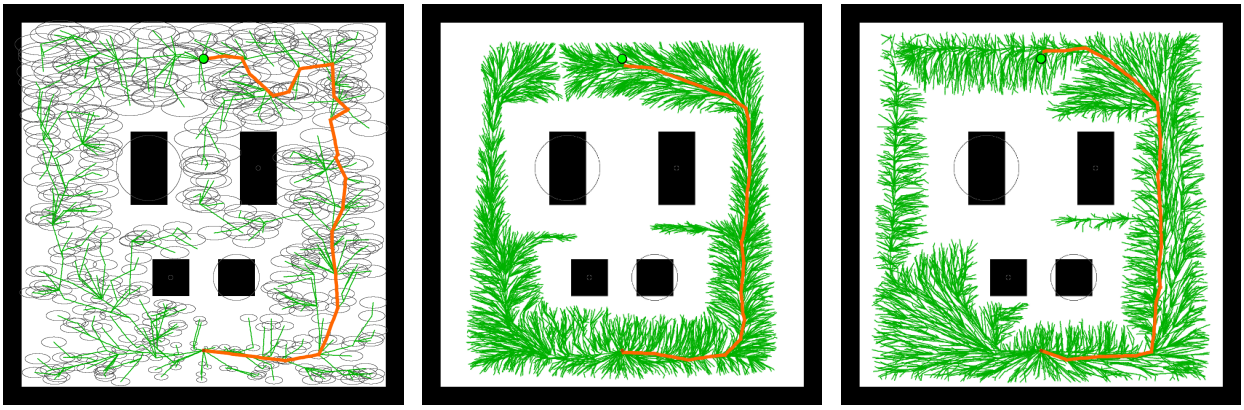


(a) $\gamma = 10^{-5}$ (b) $\gamma = 10^{0}$ (c) $\gamma = 10^{1}$



(d) $\gamma = 10^{3}$ (e) $\gamma = 10^{5}$ (f) $\gamma \to \infty$

Figure 3. Demonstrative 5000-node trees generated by CC-RRT*-Risk for various cost ratios γ for the single integrator.



(a) $2-\sigma$ uncertainty ellipses shown for tree (500 nodes) (b) $P_{x_0}^{(1)}, P_w^{(1)} \times 10$ (5000 nodes) (c) $\delta_s = 0.5$, $\delta_p = 0.9$ (5000 nodes)

Figure 4. Demonstrative trees generated by CC-RRT*-Risk for various values of δ_s, δ_p, $P_{x_0}^{(1)}$, and $P_w^{(1)}$

Table 1. Properties of solution path after 2500 nodes, over 50 trials

	Path Duration (s)				Maximum Risk Bound				Accumulated
Algorithm	Mean	SD	Min	Max	Mean	SD	Min	Max	Risk (mean)
CC-RRT*-Risk	22.7	0.40	22.1	24.2	0.002	0.002	0.001	0.013	0.004
CC-RRT*	20.3	0.14	20.0	20.6	0.189	0.010	0.158	0.200	0.782
RRT*	19.8	0.12	19.4	20.0	0.472	0.025	0.401	0.500	2.717
CC-RRT-Risk	27.2	3.46	21.4	40.5	0.143	0.047	0.029	0.200	0.421
CC-RRT	27.1	3.97	22.2	40.1	0.126	0.061	0.004	0.198	0.368
RRT	26.6	3.65	21.0	42.6	0.357	0.112	0.019	0.491	1.058

(5.5 m) wide; the upper and lower corridors are 2.5 feet (0.76 m) wide each. This environment is geometrically symmetric along the centerline of its long axis; however, the uncertainty environment is *not* symmetric. In this environment, only the bottommost obstacle has placement uncertainty,

$$c_{jt} \in \mathcal{N}(\widehat{c}_{jt}, P_{c_j}^{(2)}), \quad P_{c_j}^{(2)} = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.05 \end{bmatrix}, \quad \forall\, t;$$

all other obstacle locations are known precisely. The system is subject to the same process noise as before, $P_w^{(2)} = P_w^{(1)}$, while the localization error is 10 times as large, $P_{x_0}^{(2)} = 10P_{x_0}^{(1)}$. A time-step-wise probabilistic feasibility bound of $\delta_s = 0.8$ is enforced. A path is considered to reach the goal if the final position is within 0.5m of the goal location.

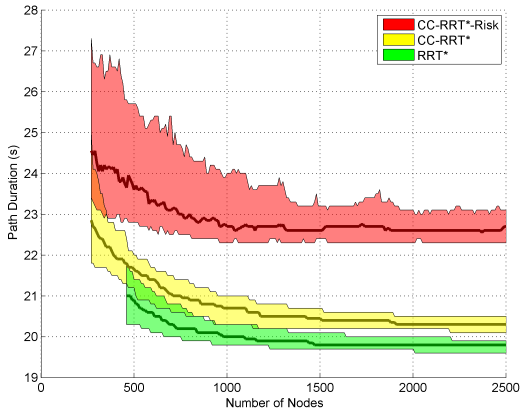
Fifty simulations were performed of each algorithm, each growing a tree of 2500 nodes. Three quantities were evaluated for the lowest-cost path, corresponding to the three components of (48): (1) path duration; (2) maximum risk bound, *i.e.,* $\max_{i=\{0,\ldots,t\}} \Delta_t(\widehat{x}_t, P_t)$; and (3) accumulated risk, defined as $\Delta_A \approx dt \sum_{t=0}^{t_f} \Delta_t(\widehat{x}_t, P_{x_t})$ (this is distinct from the path-wise risk bound (37)).

Figure 5 charts the evolution of each of these properties as a function of the number of tree nodes. In each figure, the median over all 50 trials for each algorithm is indicated as a solid line, while the shaded region surrounding it denotes the 10th-to-90th percentiles. The percentiles are not displayed when comparing the RRT-based algorithms, where the huge variation in these properties otherwise obscures the overall trends. Data is only shown once all trials have found at least one feasible path to goal, a quantity discussed further below. Table 1 gives additional statistical properties of the solution path after 2500 nodes for each algorithm.

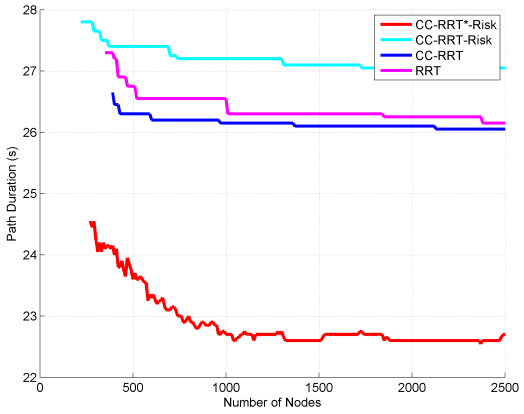
RRT* consistently achieves shorter path durations than CC-RRT* and CC-RRT*-Risk, since it is not subject to the same robustness requirements (Figure 5(a)). CC-RRT* also consistently achieves shorter path durations than CC-RRT*-Risk, since CC-RRT* only seeks to minimize path durations, while CC-RRT*-Risk includes it as one term in a multi-objective optimization. For the same reason, CC-RRT*-Risk's path duration does not decrease monotonically like CC-RRT* and RRT*, though it trends in the same direction.

On the other hand, CC-RRT*-Risk is able to identify much shorter paths than any of the RRT-based algorithms, including RRT, which does not have to consider robustness requirements (Figure 5(b)). This is largely due to the significant variation in path duration returned by the RRT-based algorithms: Table 1 notes that the standard deviation of all RRT-based algorithms is about an order of magnitude larger than their RRT*-based counterparts, while the maximum path durations returned are about twice as large.

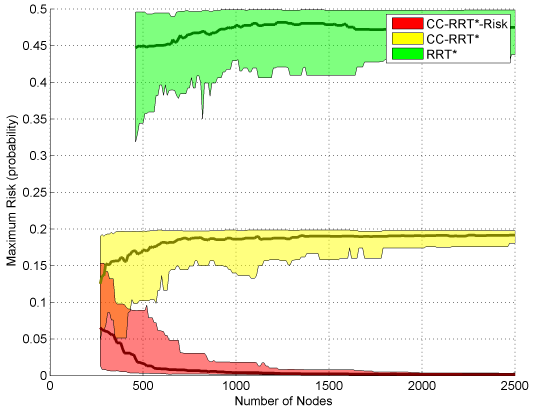
From Table 1, the final solution path returned by RRT* is consistently the shortest; its worst-case path duration is equal to or lower than the path durations returned in all other algorithm trials. However, the mean path duration of CC-RRT* is only 2.5% larger than RRT*, despite additionally



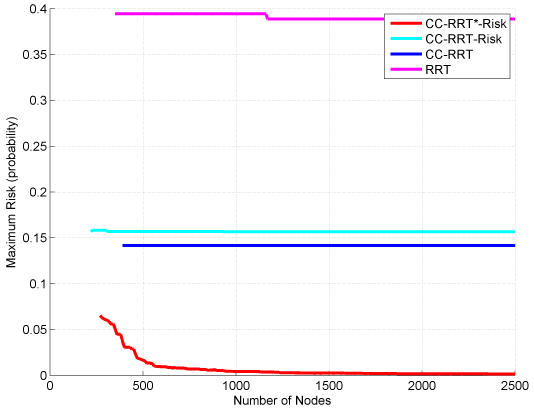
(a) Path Duration, RRT*-based algorithms



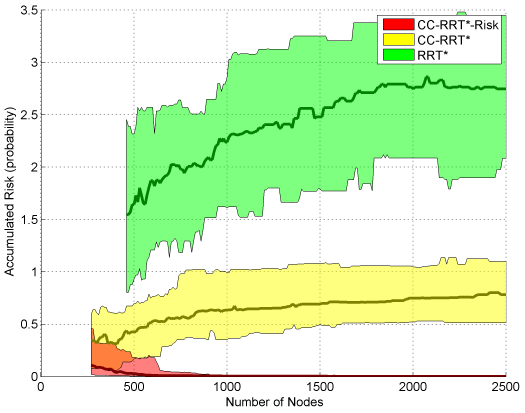
(b) Path Duration, RRT-based algorithms



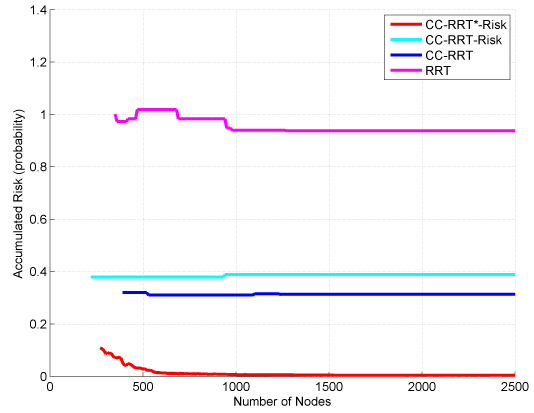
(c) Maximum Risk Bound, RRT*-based algorithms



(d) Maximum Risk Bound, RRT-based algorithms



(e) Accumulated Risk, RRT*-based algorithms



(f) Accumulated Risk, RRT-based algorithms

Figure 5. Evolution of path duration, maximum risk bound, and accumulated risk, as a function of number of nodes, for each algorithm over 50 trials. Median value is indicated as a solid line; the shaded region denotes the 10th-to-90th percentiles (not shown for right-hand-side figures).

enforcing robustness guarantees via δ_s. The paths returned by CC-RRT*-Risk are only slightly longer; its mean path duration is 15% larger than RRT* and 12% larger than CC-RRT*.

In terms of the maximum risk bound, the clearest distinction between the RRT*-based algorithms is that maximum risk tends to *increase* with more nodes for RRT* and CC-RRT*, while it tends to *decrease* with more nodes for CC-RRT*-Risk (Figure 5(c)). Because RRT* and CC-RRT* do not include any risk-based terms in their cost objectives, the rewiring process will leverage any margin between the current maximum risk value and the allowable bound to decrease path duration. As more nodes are added, the path is brought closer to obstacles, and the maximum risk value tends to converge to its bound. For CC-RRT*, that bound is $1 - \delta_s = 0.2$; for RRT*, that bound approaches 0.5, *i.e.,* the value of the risk bound if the state distribution mean were exactly on a constraint boundary. CC-RRT*-Risk, however, quickly drives the maximum risk to nearly zero, with very little variation past 1000 nodes, even though any value less than 0.2 would satisfy the probabilistic constraints.

Unlike the RRT*-based algorithms, the RRT-based algorithms (Figure 5(d)) show very little change in their risk as more nodes are added. This is due to the non-optimal nature of RRT-based algorithms, where the placement of the initial tree samples has a disproportionate impact on the nature of paths generated. Notably, CC-RRT-Risk (which uses risk terms in its objective) performs slightly worse than CC-RRT, suggesting that adding risk-based terms to the objective is only useful in an asymptotically optimal algorithm.

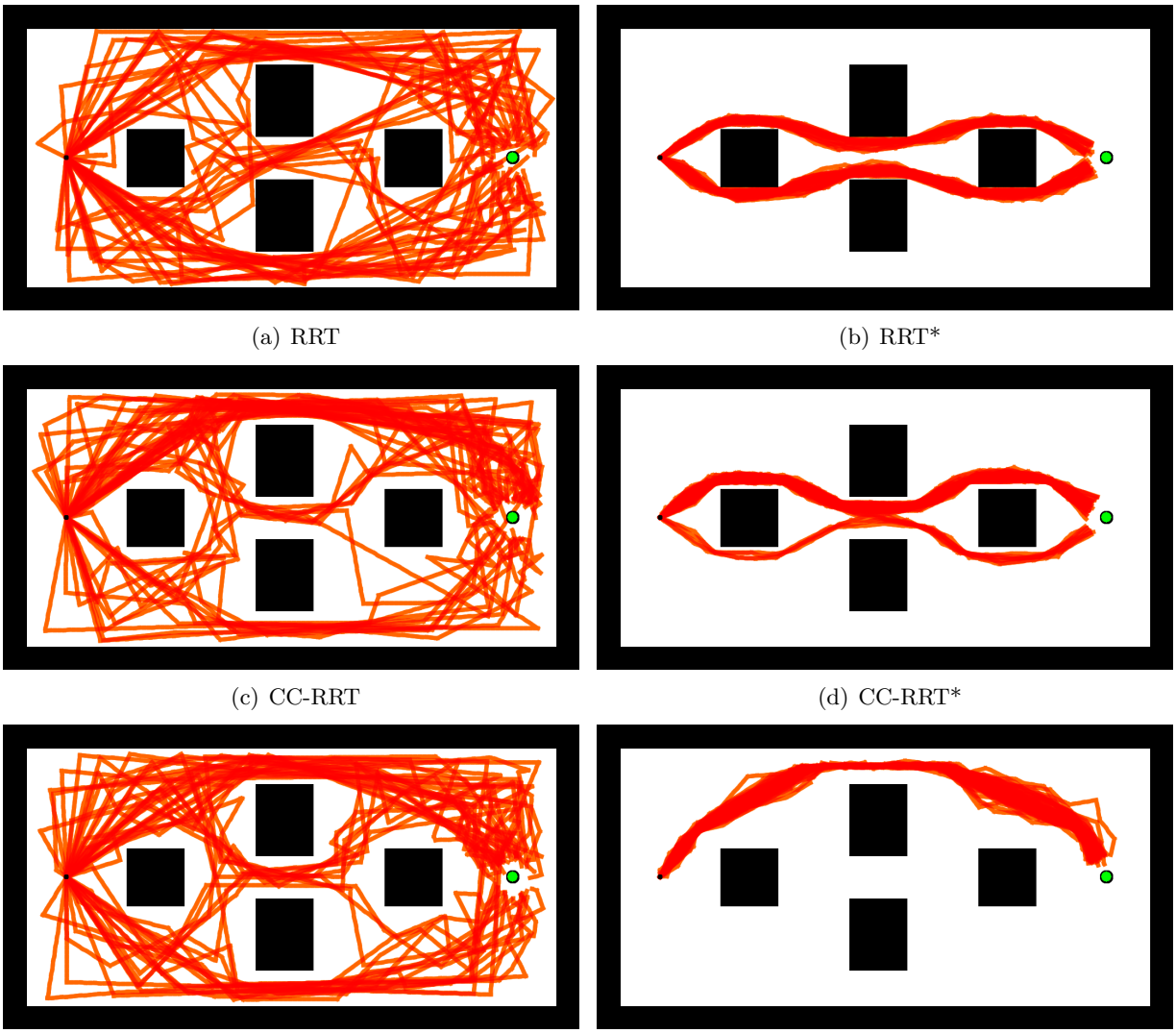
The mean and worst-case values of the maximum risk bound are 1–2 orders of magnitude smaller for CC-RRT*-Risk than all other algorithms (Table 1). As is the case for path duration, the standard deviation of all RRT-based algorithms is larger than their RRT*-based counterparts. For each algorithm except CC-RRT*-Risk, at least one trial brings the maximum risk bound very close to its allowable bound of 0.2 (CC-RRT, CC-RRT-Risk, CC-RRT*) or 0.5 (RRT, RRT*). Finally, both CC-RRT* and CC-RRT*-Risk have significantly less variation in the maximum risk bound than RRT*. The trends for accumulated risk (Figures 5(e), 5(f)) are similar.

Figure 6 shows an overlay of the final solution paths returned by each algorithm after 2500 nodes over the same 50 trials for this scenario (initial state on left, goal on right). The non-optimal algorithms (Figures 6(a), 6(c), and 6(e)) vary wildly in the qualitative nature of the paths returned, as suggested by their large corresponding variations in Table 1. There is a slight decrease in the number of paths which pass near the uncertain bottommost obstacle for CC-RRT (Figure 6(c) and CC-RRT-Risk (Figure 6(e)), which are risk-aware, compared to RRT (Figure 6(a)), which is not, but highly variable behavior is still displayed.

The final solution paths returned by the asymptotically optimal algorithms (Figures 6(b), 6(d), and 6(f)) are much more smooth and consistent; however, they do not all necessarily fall into the same homotopies. All of the paths returned by the RRT* algorithm come very close to the boundaries of the leftmost and rightmost obstacles, with very little variation within each homotopy (Figure 6(b)). However, the paths are split between passing those obstacles from above (19 out of 50) or from below (31 out of 50). Because the environment is geometrically symmetric, and RRT* is not a risk-aware algorithm, RRT* is likely to consider paths in both homotopies, even though one is subject to a much higher risk of collision than the other.

CC-RRT* is much less likely to select paths which pass near the uncertain obstacle: 41 of 50 trials pass the leftmost and rightmost obstacles from above (Figure 6(d)). Additionally, all paths chosen maintain sufficient distance from the obstacles, especially the bottommost obstacle, to ensure probabilistic feasibility constraints are satisfied. Regardless, the paths returned by the algorithm are not consistent: 3 of the 50 trials pass the leftmost and rightmost obstacles from below, while the remaining 6 trials alternate.

CC-RRT*-Risk, which not only acknowledges the risk posed by uncertainty but explicitly optimizes against it via rewiring, consistently identifies paths in the uppermost homotopy in all 50 trials



(a) RRT (b) RRT*

(c) CC-RRT (d) CC-RRT*

(e) CC-RRT-Risk (f) CC-RRT*-Risk

Figure 6. Overlay of final solution paths returned in all 50 trials for each algorithm.

Table 2. Number of nodes sampled until feasible path to goal found (rounded up to nearest 10 samples), and per-node computation results, over 50 trials

Algorithm	Nodes to Feas. Mean	Nodes to Feas. Max	Runtime per Node (ms)
CC-RRT*-Risk	73	270	17.84
CC-RRT*	80	270	17.04
RRT*	91	460	7.17
CC-RRT-Risk	78	220	1.36
CC-RRT	89	390	1.36
RRT	88	350	0.83

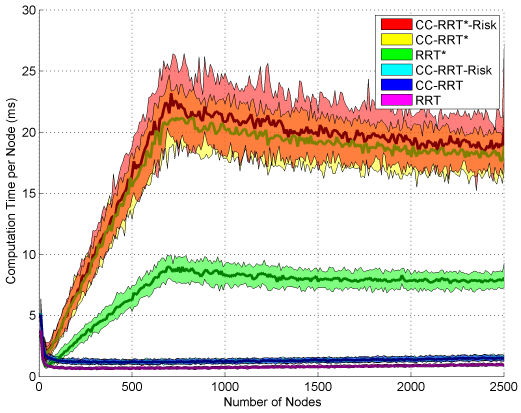


Figure 7. Evolution of computation per node, as a function of number of nodes, over 50 trials.

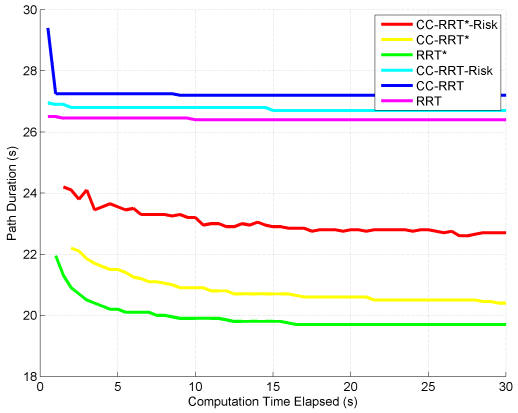


Figure 8. Evolution of mean path duration, as a function of computation time, over 50 trials.

(Figure 6(f)). It is the only algorithm of the six to recognize the asymmetry of this environment due to its uncertainty in all trials. In parts of the environment where the risk is relatively low, CC-RRT*-Risk exhibits more variation in its paths than seen from either RRT* or CC-RRT*. But as the system passes the uppermost obstacle, there is very little variation in the paths at all: all 50 paths pass very near the centerline of that corridor. Because CC-RRT*-Risk incorporates risk bounds within its objective, regions with higher risk correspond to larger cost gradients, and thus less variation.

Table 2 shows the mean and maximum number of nodes required to find a feasible path to the goal region $\mathcal{X}_{\text{goal}}$ across all trials for each scenario considered above, rounded up to the nearest 10 samples. The number of nodes required is fairly consistent across all algorithms: a feasible path is found on average in under 100 nodes, while a feasible path is always found within 500 nodes. This implies that the effect of these modifications on finding feasible paths is limited, at best.

Table 2 also shows the average time required to generate a feasible node for each algorithm across all trials, including failed attempts to connect new samples[b]. The most notable increase in computation comes from switching from an RRT-based algorithm to its RRT*-based equivalent, resulting in a per-node runtime increase by a factor of 9–13. This factor can be significantly affected by the complexity of the dynamics and the steering law that is used, as RRT-based algorithms do not require use of a steering law, and thus must be considered carefully. By comparison, introducing robustness via chance constraints only results in a runtime increase by a factor of 1.6–2.5 per node, consistent with previous results.[4] The impact of introducing risk-based terms into the cost function is minimal. Regardless, both CC-RRT*-Risk and CC-RRT* remain suitable for real-time use.

Figure 7 shows an evolution of the computation time required per node, as a function of tree size; computation has been averaged every 10 nodes to smooth the plots. The computation required per node for the RRT*-based algorithms increases at a faster rate up to about 700 nodes or so, where it begins to level off and even decrease slightly. This is likely due to the radius r_n of (46) decreasing over time, limiting the number of rewiring connections attempted.

Finally, 50 more trials were performed for each algorithm in which each Algorithm 3 is given 30 seconds to grow a tree, with no upper bound on the number of nodes. Figure 8 shows the evolution of the mean path duration, as a function of computation time elapsed, for these trials. Even with the robustness modifications of CC-RRT* and CC-RRT*-Risk, both algorithms find higher-quality (in terms of past duration) than any RRT-based algorithms, with robustness guarantees, within the initial seconds of computation.

[b] All simulations were performed in a Java implementation on a quad-core 2.40-GHz processor.

C. Double Integrator

Consider the more complex 2D double integrator dynamics,

$$x_{t+1} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} \frac{1}{2}dt^2 & 0 \\ 0 & \frac{1}{2}dt^2 \\ dt & 0 \\ 0 & dt \end{bmatrix} u_t + w_t,$$

where, again, $dt = 0.1$ s; the system operates in the same environment as Section VII-B. In addition to the previous state constraints, this problem is also subject to the velocity state constraints $|v_t^x| \leq v_{\max},\ |v_t^y| \leq v_{\max}$, where $v_{\max} = 2.5$ m/s, as well as the input $u_t = (a_t^x, a_t^y)$ constraints $|a_t^x| \leq \bar{a},\ |a_t^y| \leq \bar{a}$, where $\bar{a} = 5.0$ m/s^2.

The system is subject to the same three forms of uncertainty, though their values have changed:

$$P_{x_0}^{(3)} = 10^{-3} \begin{bmatrix} I_2 & 0_2 \\ 0_2 & 0_2 \end{bmatrix},\ P_w^{(3)} = 10^{-6} \begin{bmatrix} 0_2 & 0_2 \\ 0_2 & I_2 \end{bmatrix},\ P_{c_j}^{(3)} = 0.05 \begin{bmatrix} I_2 & 0_2 \\ 0_2 & 0_2 \end{bmatrix},\ \forall\, t.$$

As in the previous scenario, only the bottommost obstacle is uncertain. The system is required to satisfy $\delta_s = 0.8$; no path-wise probability bound is imposed.

The sampling strategy augments the position sampling for the single integrator with velocity sampling. With some small probability (currently 5%), the 2D velocity is sampled uniformly within the full velocity constraints $|v_t^x| \leq v_{\max},\ |v_t^y| \leq v_{\max}$, ensuring that every feasible state has a non-zero probability of being sampled. Otherwise, the 2D velocity is sampled as

$$v_x = \tilde{v}\cos\tilde{\psi}, \qquad v_y = \tilde{v}\sin\tilde{\psi},$$

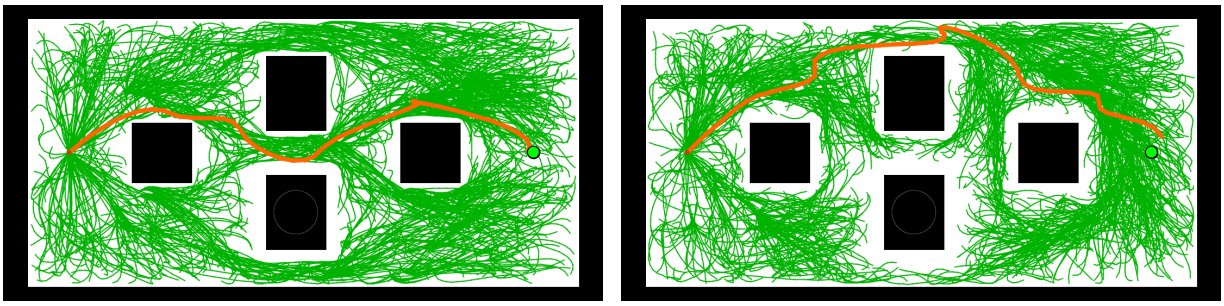
where $\tilde{\psi}$ is sampled uniformly between 0 and 2π, and $\tilde{v}$ is sampled uniformly between $V_{\min} = 1.0$ m/s and $V_{\max} = 2.0$ m/s. The steering law fits a cubic spline to each coordinate in order to maintain an approximate traversal speed $\bar{v} = 1.0$ m/s; the details are omitted for brevity. A path is considered to reach the goal if the final position is within 0.5m of the goal location. The nearby node function uses a maximum radius $\mu = 5$ m, while $\mu(\mathcal{X}_{\text{free}}) = 4v_{\max}^2 A_{\text{free}}$.

Figure 9 demonstrates typical trees and final solution paths generated by CC-RRT* and CC-RRT*-Risk for this scenario. Similar behaviors are observed for both algorithms as in the single-integrator scenarios, though the degree of suboptimality in the solution paths is higher due to both the reduced number of nodes and the increased state dimension. Regardless, all paths in each tree satisfy probabilistic feasibility requirements for $\delta_s = 0.8$, with additional conservatism induced in the solution path by using a risk-based objective in Figure 9(b).

Finally, Figure 10 shows a 1000-node CC-RRT*-Risk tree and solution path generated for the double integrator dynamics (subject to the same localization error and process noise) in a cluttered 20m $\times$ 10m environment, consisting of 30 obstacles with randomized placement uncertainty. This demonstrates the scalability of the CC-RRT* algorithm to very complex environments, in both the number of obstacles and the uncertainty characterization.

VIII. Conclusions

This paper has introduced the CC-RRT* algorithm, for robust, scalable, and asymptotically optimal path planning. The algorithm efficiently computes bounds on risk of constraint violation using a chance constraint formulation – expanded in this work to consider path-wise feasibility bounds and probabilistic environmental boundaries – to ensure all trajectories considered are both



(a) CC-RRT* (b) CC-RRT*-Risk

Figure 9. Demonstrative 1500-node trees and minimum-cost paths generated by the CC-RRT* algorithms for the double integrator.

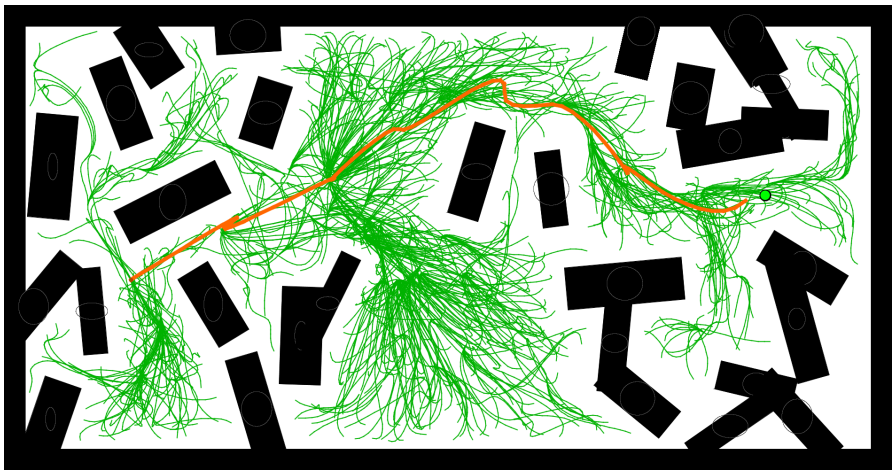


Figure 10. Demonstrative 1000-node tree and minimum-cost path generated by the CC-RRT*-Risk algorithm for the double integrator in a highly-cluttered and uncertain environment.

dynamically and probabilistically feasible. These risk bounds are also utilized within a novel, risk-based cost function, shown to be admissible for RRT*, such that the RRT* framework can be leveraged to ensure asymptotic optimality of paths returned. Simulation results have demonstrated that CC-RRT* can be utilized to identify smooth, robust trajectories, displaying a level of risk-averse behavior specified by the user.

Future work will expand the theory behind CC-RRT*, including extension to more complex cost functions, nonlinear dynamics, and/or non-Gaussian uncertainty.[28] Additional simulation and hardware results will be explored for more complex scenarios and dynamics, including non-holonomic vehicles;[29] dynamic obstacles, including pursuit-evasion scenarios;[30] and dynamic real-time operations.

References

[1]K. Iagnemma and M. Buehler, eds. Special issue on the DARPA grand challenge, part 1. *Journal of Field Robotics*, 23(8):461–652, August 2006.

[2]L. Blackmore, H. Li, and B. Williams. A probabilistic approach to optimal robust path planning with obstacles. In *American Control Conference (ACC)*, 2006.

[3]S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Iowa State University, October 1998.

[4]B. Luders, M. Kothari, and J. P. How. Chance constrained RRT for probabilistic robustness to environmental uncertainty. In *AIAA Guidance, Navigation, and Control Conference (GNC)*, Toronto, Canada, August 2010. (AIAA-

2010-8160).

[5]G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots*, 35(1):51–76, 2013.

[6]Y. Kuwata, J. Teo, G. Fiore, S. Karaman, E. Frazzoli, and J. P. How. Real-time motion planning with applications to autonomous urban driving. *IEEE Transactions on Control Systems Technology*, 17(5):1105–1118, September 2009.

[7]S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, June 2011.

[8]S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning. In *Robotics: Science and Systems (RSS)*, 2010.

[9]S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[10]S. M. LaValle and R. Sharma. On motion planning in changing, partially-predictable environments. *International Journal of Robotics Research*, 16(6):775–824, 1995.

[11]M. Ono and B. C. Williams. Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint. In *Proceedings of the IEEE Conference on Decision and Control*, 2008.

[12]L. Blackmore and M. Ono. Convex chance constrained predictive control without sampling. In *Proceedings of the AIAA Guidance, Navigation and Control Conference*, 2009.

[13]M. Ono, L. Blackmore, and B. C. Williams. Chance constrained finite horizon optimal control with nonconvex constraints. In *Proceedings of the American Control Conference*, 2010.

[14]M. P. Vitus and Tomlin. C. J. A hybrid method for chance constrained control in uncertain environments. In *IEEE Conference on Decision and Control (CDC)*, 2012.

[15]L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, August 1996.

[16]P. E. Missiuro and N. Roy. Adapting probabilistic roadmaps to handle uncertain maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1261–1267, Orlando, FL, May 2006.

[17]L. J. Guibas, D. Hsu, H. Kurniawati, and E. Rehman. Bounded uncertainty roadmaps for path planning. In *Proceedings of the International Workshop on the Algorithmic Foundations of Robotics*, 2008.

[18]B. Burns and O. Brock. Sampling-based motion planning with sensing uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3313–3318, Roma, Italy, April 2007.

[19]R. Alterovitz, T. Siméon, and K. Goldberg. The stochastic motion roadmap: A sampling framework for planning with Markov motion uncertainty. In *Proceedings of Robotics: Science and Systems*, 2007.

[20]V. A. Huynh, S. Karaman, and E. Frazzoli. An incremental sampling-based algorithm for stochastic optimal control. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2865–2872, Saint Paul, MN, 2012.

[21]S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 28(11-12):1448–1465, 2009.

[22]A. Agha-mohammadi, S. Chakravorty, and N. M. Amato. FIRM: Feedback controller-based information-state roadmap – a framework for motion planning under uncertainty –. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4284–4291, San Francisco, CA, 2011.

[23]N. A. Melchior and R. Simmons. Particle RRT for path planning with uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[24]G. Kewlani, G. Ishigami, and K. Iagnemma. Stochastic mobility-based path planning in uncertain environments. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1183–1189, St. Louis, MO, USA, October 2009.

[25]R. Pepy, M. Kieffer, and E. Walter. Reliable robust path planning. *International Journal of Applied Math and Computer Science*, 1:1–11, 2009.

[26]J. van den Berg, P. Abbeel, and K. Goldberg. Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information. *International Journal of Robotics Research*, 30(7):1448–1465, 2011.

[27]A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2011. To appear.

[28]B. Luders and J. P. How. Probabilistic feasibility for nonlinear systems with non-Gaussian uncertainty using RRT. In *AIAA Infotech@Aerospace Conference*, St. Louis, MO, March 2011. (AIAA-2011-1589).

[29]S. Karaman and E. Frazzoli. Sampling-based optimal motion planning for non-holonomic dynamical systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5026–5032, Karlsruhe, Germany, 2013.

[30]S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for a class of pursuit-evasion games. In *Workshop on the Algorithmic Foundations of Robotics*, Atlanta, GA, 2010.