IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Koopman Operator Method for Chance-Constrained Motion Primitive Planning

Geordan Gutow
George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0405 USA
Email: geordangutow@gatech.edu

Jonathan D. Rogers
Daniel Guggenheim School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332-0405 USA
Email: jonathan.rogers@ae.gatech.edu

***Abstract*— The use of motion primitives to plan trajectories has received significant attention in the robotics literature. This work considers the application of motion primitives to path planning and obstacle avoidance problems in which the system is subject to significant parametric and/or initial condition uncertainty. In problems involving parametric uncertainty, optimal path planning is achieved by minimizing the expected value of a cost function subject to probabilistic (chance) constraints on vehicle-obstacle collisions. The Koopman operator provides an efficient means to compute expected values for systems under parametric uncertainty. In the context of motion planning, these include both the expected cost function and chance constraints. This work describes a maneuver-based planning method that leverages the Koopman operator to minimize an expected cost while satisfying user-imposed risk tolerances. The developed method is illustrated in two separate examples using a Dubins car model subject to parametric uncertainty in its dynamics or environment. Prediction of constraint violation probability is compared with a Monte Carlo method to demonstrate the advantages of the Koopman-based calculation.**

I. Introduction

Numerous problems in robotics require planning a trajectory over a finite temporal or spatial horizon. Motion primitives, originally formalized by Frazzoli *et al.* [1], are one technique for solving this problem. The primary advantage of motion primitives is that dynamically feasible trajectories (primitives) are computed offline and stored, avoiding the need to simulate dynamics or solve optimal control problems in real-time. Primitives are chained together to form paths, allowing rapid calculation of the state trajectory and accumulated cost. Motion primitives have been successfully applied to a wide variety of path planning problems to date [2]–[6]. The technique is particularly useful when the dynamics are expensive to simulate as in [2] and [3].

Motion primitive planning becomes significantly more complicated when considering systems with parametric uncertainty. A key assumption in the formulation of motion primitive planning is that the state evolution from the beginning to the end of each maneuver is, up to an offset in the cyclic coordinates, the same every time the maneuver is executed. Under parametric uncertainty in the system dynamics, this may no longer be true; the state after executing a maneuver may depend on the unknown parameters. Despite this difficulty, several authors have proposed extensions to motion primitive planning to address uncertainty. Schouwenaars *et al.* [7] proposed a robust motion primitive algorithm that used Monte Carlo simulation to estimate the maximum deviation in state values that could result from each maneuver. Majumdar and Tedrake [8] developed a robust motion primitive-like planner in which sum-of-squares optimization was used to compute invariant funnels that provide zero-collision guarantees. While robust approaches are useful in many cases, they offer no mechanism to trade probability of constraint violation for better system performance. In many cases, it may be possible to obtain a better path (in terms of overall cost or score) if constraint violation is allowed with non-zero probability. As an example, Thakur *et al.* [2] use motion primitives in a Markov Decision Process with massively parallel GPU computation to estimate transition probabilities, computing a probabilistically optimal path under uncertainty.

Despite this limited work, motion primitive planning under parametric uncertainty remains a highly under-explored area. Interestingly, no prior work (to the authors' knowledge) has investigated the possibility of chance-constrained planning with motion primitives. In chance-constrained planning, the probability of constraint violation is computed and constrained to be less than some risk tolerance. The advantages of the chance-constrained approach over robust techniques or cost-penalized approaches are well-known [9]–[12]. The current paper seeks to fill this gap in the motion primitive literature by presenting a technique for chance-constrained motion primitive planning for problems involving parametric uncertainty.

Path planning under parametric uncertainty (equivalently, initial condition uncertainty) requires some means of *uncertainty quantification* (UQ) to evolve the state PDF along the path, so that the expected value of the cost and constraint violation probabilities may be calculated. A standard technique is Monte Carlo sampling, used in [2], [12], [13]. It has the distinct advantages of application to nonlinear systems, arbitrary probability distributions, and unlimited simulation horizons, giving it wider applicability than analytical or parametric techniques like those used in [10], [14]–[16]. From an implementation perspective, Monte Carlo methods also benefit greatly from massively parallel computation on GPUs [2] [17]. However, Monte Carlo scales poorly to higher dimensions and the underlying PDF must be estimated from

the evolved samples. As discussed in [18], a variety of techniques exist to alleviate this in special cases.

The Frobenius-Perron (FP) operator and its adjoint the Koopman operator form the basis of alternative "explicit UQ" techniques for parametric uncertainty that have been shown in [18]–[22] to scale better than Monte Carlo while preserving much of the generality and parallelizability of the technique. In particular, Meyers *et al.* [21] showed that the Koopman operator can be used to compute constraint violation probabilities and expected costs more accurately than Monte Carlo at a given number of samples, while providing tractable integration domains. Leonard *et al.* [22] used GPUs and Lobachevsky spline integration to apply the Koopman approach in a probabilistic airdrop optimization problem involving five dimensions of parameter uncertainty.

The Koopman operator provides an infinite dimensional linear encoding of the state evolution of nonlinear dynamical systems. Existing work has shown that truncations of basis function expansions of the operator provide an effective means of system linearization that is amenable to discovery from system state data [23]–[28]. This provides an approximate continuous representation of the evolution of an observable of the system state. In the present approach the Koopman operator is instead obtained "exactly" (up to integration error) at a discrete set of selected points in the state space via the method of characteristics [21]. Basis function expansion and data-driven learning are avoided by relying on an existing (potentially black-box) model and by sacrificing information about the evolution of the observable of the state of the system at unaddressed points.

This paper proposes a novel technique for motion primitive planning under parametric uncertainty using the Koopman operator. The uncertainty space is discretized offline and the state evolution along each primitive is computed at each point and stored. Online, the probability of constraint violation for each stored trajectory is obtained and pulled back to the parameter uncertainty domain (this is the action of the Koopman operator). A simple expected value calculation provides the total constraint violation probability. Two path planning examples are shown for a Dubins car model under parametric uncertainty, involving various types of chance constraints. The accuracy of the constraint violation probability predictions is compared with Monte Carlo methods to demonstrate the computational advantages of the proposed approach. The planning algorithm described here can be viewed as a particular application of the Koopman operator approach to probabilistic optimization proposed in [21].

II. Mathematical Background

Consider a time-invariant dynamical system $\mathbb{D}$ with state space $x \in \mathbb{X}$ subject to control input $u \in \mathbb{U}$ with parameter vector $s \in \mathbb{S}$, governed by the differential equation $\dot{x} = f(x, u, s)$. Let s be uncertain with a known probability distribution $P(s)$. Furthermore let $\rho_\mu(x_0, t, s)$ be the state of the system after evolving for time t from x_0 under the piecewise-continuous control law $\mu : [0, t_f] \to \mathbb{U}$ when the parameter vector takes on value s. Equip $\mathbb{X}$ with a Lie group $\mathbb{G}$ and a group action $\Psi : \mathbb{G} \times \mathbb{X} \to \mathbb{X}$. For $g \in \mathbb{G}$, the group action $\Psi(g, x)$ will be written as $g \circ x$. Suppose that the dynamical system is invariant with respect to Ψ in the sense of [1]: $\forall (g \in \mathbb{G}, x_0 \in \mathbb{X}, t \in [0, t_f], \mu : [0, t_f] \to \mathbb{U}, s \in \mathbb{S}), g \circ \rho_\mu(x_0, t, s) = \rho_\mu(g \circ x_0, t, s)$. Then, $\mathbb{G}$ is a symmetry group of $\mathbb{D}$. Accordingly, $\mathbb{G}$ divides $\mathbb{X}$ into equivalence classes via the relation $x_1 \cong x_2 \iff \exists g \in \mathbb{G}$ s.t. $x_1 = g \circ x_2$.

A cyclic coordinate is a (generalized) coordinate on which the system Lagrangian does not depend. The other coordinates are non-cyclic coordinates. The dynamics of $\mathbb{D}$ are invariant to changes in the cyclic coordinates, which are the very same coordinates acted on by a symmetry group of $\mathbb{D}$.

A. Maneuvers

As described in [1], a deterministic maneuver π is a triple $(\text{Pred}(\pi), g_\pi, \text{Succ}(\pi))$ containing the equivalence class of states from which the maneuver may be executed ($\text{Pred}(\pi)$), the group displacement after maneuver execution (g_π), and the equivalence class of states in which the maneuver terminates ($\text{Succ}(\pi)$). In order to address constraints (or integrated costs) that apply during a maneuver, such as obstacle avoidance, g_π is modified to be the group displacement as a function of τ, the time elapsed since the maneuver began. However, this work will consider only constraints and integrated costs that do not depend on the non-cyclic coordinates, so the time history of the non-cyclic coordinates need not be stored.

The introduction of parametric uncertainty transforms $\text{Succ}(\pi), g_\pi(\tau)$ into random variables. Instead of a unique $\text{Succ}(\pi)$ and $g_\pi(\tau)$, a maneuver possesses a family of successor equivalence classes $\text{Succ}(\pi, s)$ and group displacement histories $g_\pi(\tau, s)$, each parameterized by s. The trajectory resulting from a maneuver π executed from a particular state and time is given by the function $H_\pi : \mathbb{X} \times \mathbb{R} \times \mathbb{S} \to \mathbb{T}$, where $\mathbb{T}$ is the set of trajectories (functions $\mathbb{R} \to \mathbb{X}$) of $\mathbb{D}$.

B. The Chance-Constrained Maneuver Planning Problem

For a current state x_0, current time t_0, and set of available maneuvers Π, the maneuver π^* minimizes the expected cost of the resulting state trajectory $x_{\pi^*} \in \mathbb{T}$ while ensuring that the total probability of violating a constraint is below the risk tolerance r (a slightly more general formulation could impose different risk tolerances for each constraint, making r a vector). The cost function $J : \mathbb{T} \to \mathbb{R}$ can include both an integrated and terminal cost. It is assumed that the violation of one constraint is independent of the violation of any other constraint. Under these assumptions the optimization problem can be formulated over the set $\Pi(t_0, x_0)$ of available maneuvers as:

$$\underset{\pi \in \Pi(t_0, x_0)}{\operatorname{argmin}} \; E[J(H_\pi(x_0, t_0, s))] \text{ s.t.} \quad (1)$$
$$1 - \prod_i \left(1 - E\left[C_i\left(H_\pi(x_0, t_0, s)\right)\right]\right) \leq r$$

Here, $C_i : \mathbb{T} \to [0, 1]$ is the ith constraint indicator function. C_i returns the probability that the ith constraint is violated

during the trajectory. In the case of a typical obstacle, this function returns 1 if the trajectory intersects the obstacle and 0 otherwise. As $\Pi(t_0, x_0)$ is finite, this optimization can be solved by evaluating the relevant expectations for each $\pi \in \Pi(t_0, x_0)$.

C. Expectations via the Koopman Operator

Consider a function $h_t : \mathbb{X} \to \mathbb{R}$, an observable of the state at time t. A family of Koopman operators $\mathbb{K}_\pi(t)$ is defined for each maneuver (one for each t):

$$\mathbb{K}_\pi(t)h_t(x) = h_t(H_\pi(x_0, t_0, s)(t)) \tag{2}$$

where $H_\pi(x_0, t_0, s)(t)$ is the state obtained by evaluating trajectory $H_\pi(x_0, t_0, s)$ at time t. For a chosen maneuver, the Koopman operator acts to map a function of the system state at some future time to a function of the initial conditions and uncertain parameters. The right hand side of (2) can be computed at freely chosen initial condition/uncertain parameter points by direct simulation of the dynamics to time t and evaluation of h_t on the resulting state. For a careful derivation of the Koopman operator pull-back, see [21].

Constraint indicator functions and the most general form of the cost are in fact functions of the entire state trajectory rather than a particular time step, and so cannot be handled directly. This is resolved by augmenting the state vector as shown below for an integrated cost term. Constraint indicator functions can be handled similarly. Suppose $J(H_\pi(x_0, t_0, s)) = \int_0^{\tau_f(\pi,s)} d(t, H_\pi(x_0, t_0, s)(t))dt$ ($\tau_f(\pi, s)$ is the duration of maneuver π when the uncertainty takes on value s). The state is augmented with the accumulated value of the integrand (symbol: D) and its dynamics defined:

$$\dot{D} = d(t, x) \tag{3}$$

Then, define a function $z : \mathbb{A} \to \mathbb{R}$ (where $\mathbb{A}$ is the set of augmented states) that extracts D. As long as d satisfies the conditions for existence and uniqueness as discussed in [21], the Koopman operator may still be used to pull-back z. By restricting to cost functions including only terminal costs plus a sufficiently smooth integral cost, the J in (1) is thus equivalent to a function z of the augmented terminal state $\hat{x}_f$ and the expectation is computed as:

$$E[J(H_\pi(x_0, t_0, s))] = E[\mathbb{K}_\pi(t)z(\hat{x}_f)] \tag{4}$$

$$= \int_{\mathbb{S}} P(s)\mathbb{K}_\pi(t)z(\hat{x}_f)ds \tag{5}$$

In this work, the integral in (5) is approximated by gridding the uncertainty space $\mathbb{S}$ with n sample points s_j and computing $z(\hat{x}_f)$ for the trajectories corresponding to those samples:

$$E[J(H_\pi(x_0, t_0, s))] \tag{6}$$

$$\approx \sum_{j=1}^{n} P(s_j)z(H_\pi(x_0, t_0, s_j)(\tau_f(\pi, s_j))\Delta s$$

where $H_\pi(x_0, t_0, s_j)$ is here understood to be the evolution of the augmented state and Δs is the integration voxel size. As the locations of s_j may be chosen freely, more sophisticated numerical integration schemes may be used if appropriate, particularly in higher-dimensional problems. For further discussion of possible integration methodologies, see [21] and [22].

III. Methodology

A. Maneuver Representation

In order to (approximately) solve (1), H_π must be computable for any initial condition in Pred(π) and arbitrary initial time, at each sample point s_j. Since $\mathbb{D}$ is time-invariant, variable initial time simply imposes a time-offset in the resulting trajectory. As discussed in II-A, the time-varying change in cyclic coordinates is represented by $g_\pi(t, s)$, while the change in non-cyclic coordinates is relevant only at the terminal state and so is captured by storing Succ(π, s). Thus, for a particular maneuver, H_π can be computed as follows:

$$H_\pi(x_0, t_0, s_j) = g_\pi(t - t_0, s_j) \circ x_0 \tag{7}$$

The combination of $g_\pi(t, s_j)$ and Succ(π, s_j) is referred to as the s_j-realization of maneuver π. For planning purposes, an uncertain maneuver consists of a collection of realizations for values of the uncertain parameters that, collectively, cover $\mathbb{S}$. The probability of each realization, $P(s_j) \times \Delta s$, is also stored. Deterministic maneuvers can be handled in this framework via a single realization with probability one.

Maneuvers are augmented with a few precomputed values that are useful for planning purposes. The quantity $\bar{g}_\pi$ is the expected group displacement during the execution of maneuver π, defined as:

$$\bar{g}_\pi = \exp\left(\sum_{j=1}^{n} P(s_j) \exp^{-1}(g_\pi(\tau_f(\pi, s_j), s_j))\right) \tag{8}$$

where exp and $\exp^{-1}$ are the exponential and inverse exponential maps associated with $\mathbb{G}$.

The quantity $\bar{y}_\pi$ is the expected offset in the non-cyclic coordinates at the end of the maneuver. It can be obtained from the same state trajectories used to obtain $g_\pi(\tau, s)$:

$$\bar{y}_\pi = \sum_{j=1}^{n} P(s_j)(g_\pi^{-1}(\tau_f(\pi, s_j), s_j) \circ H_\pi(x_0, t_0, s_j)(\tau_f(\pi, s_j)) - x_0) \tag{9}$$

The quantities $\bar{g}_\pi$ and $\bar{y}_\pi$ together allow the immediate calculation of $\bar{x}_f(\pi)$, the expected vehicle state after the maneuver, for any π-compatible initial condition x_0:

$$\bar{x}_f(\pi) = \bar{g}_\pi \circ x_0 + \bar{y}_\pi \tag{10}$$

Lastly, $\bar{\tau}_\pi$ is the expected duration of the maneuver π:

$$\bar{\tau}_\pi = \sum_{j=1}^{n} P(s_j)\tau_f(\pi, s_j) \tag{11}$$

B. Maneuver Library Closure

Maneuver-based path planning requires that the maneuver library be "closed": the exit state of every maneuver must be a valid entry state for at least one other maneuver in the library. Introducing uncertainty into the execution of maneuvers creates difficulty, as the uncertainty can cause the volume encompassing the possible exit states to expand with each maneuver executed. In the example system considered in Section IV, this is resolved by including "recovery maneuvers" with entry states encompassing the exit states of the regular maneuvers out to ± 4 standard deviations. The recovery maneuvers provide a way to drive the non-cyclic coordinates (angular velocity in the example below) back to zero from any exit state within four standard deviations of the mean, yielding a library that is approximately probabilistically closed. This is possible because the control input directly regulates the non-cyclic coordinate. In a more complex system, closing the library will generally be more difficult. Developing methods for constructing closed libraries in the presence of complex dynamics and parametric uncertainty is left to future work.

C. Planning with Uncertain Maneuvers

A recursive depth-limited search is proposed to select the next maneuver according to Eq. (1), while accounting for the possibility of a poorly chosen maneuver placing the vehicle in an inescapable collision. The approach is outlined in procedures 1 and 2.

In procedure 1, past_cp is the collision probability accumulated along the current sequence of primitives; thus, the initial call to *choose* sets this to 0. x_{obs} is the state from which the environment is observed while t_{obs} is the time at which that observation takes place; in the initial call these are equal to the current state x_0 and time t_0.

First, the set of maneuvers available in the current state is obtained. Procedure 2 is called to compute the collision probability for each maneuver. The probability of a collision during the current maneuver is combined with the collision probability accumulated over past maneuvers (past_cp) assuming that collision in any maneuver is independent of collision in any other. Procedure 2 computes the expected state and time following the execution of the maneuver and, if the new state is a goal state or outside the planning horizon, the accumulated collision probability is returned to *choose*. Otherwise, procedure 1 is called with the "current" state and time set to the expected state and time following the maneuver under consideration. This recursion continues until the expected state leaves the planning horizon or reaches the goal, at which point the accumulated collision probability is returned. By including collision probability from maneuvers planned from the expected exit state, the procedure generates a heuristic that discourages choosing maneuvers that are safe now but leave the vehicle with no safe maneuvers later. Procedure 1 receives from its call to procedure 2 a heuristic value that describes how risky each maneuver is. From the maneuvers with risk less than the tolerance, the maneuver with the lowest cost is selected; if no maneuvers have acceptable risk, the least risky maneuver is chosen.

Procedure 1 choose(past_cp, $x_{\text{obs}}, t_{\text{obs}}, x_0, t_0$)

$\Pi \leftarrow$ set of maneuvers compatible with x_0
cp $\leftarrow$ {evaluate(past_cp, π, x_{obs}, t_{obs}, x_0, t_0) : $\pi \in \Pi$}
safe_maneuvers $\leftarrow$ {$\Pi[i]$: cp[i] $< r$ }
if safe_maneuvers is empty **then**
  safest_maneuver_id $\leftarrow \operatorname{argmin}$ cp
  best_maneuver $\leftarrow \Pi$[safest_maneuver_id]
  best_cp $\leftarrow$ cp[safest_maneuver_id]
else
  costs $\leftarrow$ {$\sum_{j=1}^{n} P(s_j) J(H_\pi(x_0, t_0, s_j))\Delta s$: $\pi \in$ safe_maneuvers}
  bmi $\leftarrow \operatorname{argmin}$ costs
  best_maneuver $\leftarrow$ safe_maneuvers[bmi]
  best_cp $\leftarrow$ cp[index of best_maneuver in Π]
return best_cp, best_maneuver

Procedure 2 evaluate(past_cp, π, x_{obs}, t_{obs}, x_0, t_0)

cp $\leftarrow 1 - \prod_i (1 - \sum_{j=1}^{n} P(s_j) C_i(H_\pi(x_0, t_0, s_j))\Delta s)$
total_cp $\leftarrow 1 - (1 - \text{past\_cp}) * (1 - \text{cp})$
$x_e \leftarrow \bar{g}_\pi \circ x_0 + \bar{y}_\pi$
$t_e \leftarrow t_0 + \bar{\tau}_\pi$
if x_e is a goal **or** $x_e \notin$horizon(x_{obs}) **then**
  return total_cp
else
  cp,_ $\leftarrow$choose(total_cp,$x_{\text{obs}}, t_{\text{obs}}, x_e, t_e$)
  return cp

Instead of this heuristic, the actual probability of collision accounting for future decisions could be computed. This would require calling procedure 1 from each possible exit state of the maneuver being considered, then taking an expectation across the results. The computational complexity of this is $O(m^k n^k)$ where m is the (average) number of maneuvers available at each stage, n is the number of samples used to cover the uncertainty space, and k is the search depth. The heuristic's complexity is only $O(m^k n)$, a dramatic improvement since n is generally large. The heuristic's accuracy depends on how widely spread the exit states are, which could be quantified by computing variances. A more sophisticated planner might improve performance in high risk situations by doing the full calculation when the variance is large; this development is left to future work.

IV. Example System and Maneuvers

The planner is demonstrated on a Dubins car-like vehicle subject to different forms of model or environmental parametric uncertainty. The vehicle travels at constant speed v in the x, y plane, with state $[x, y, \theta, \omega]^T$ and motion governed

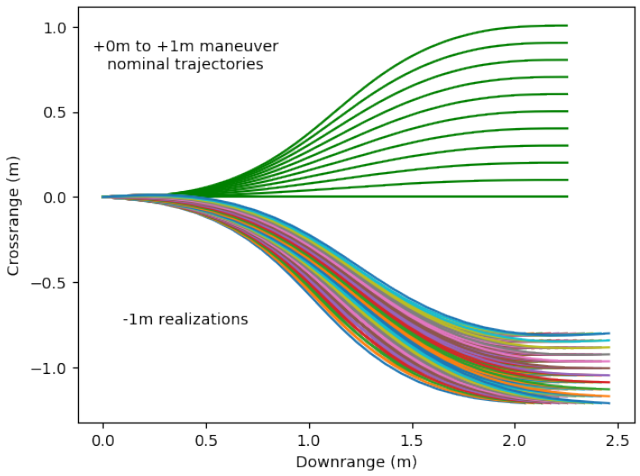

Fig. 1. Top: Nominal trajectories for 11 of the 21 maneuvers (negative maneuvers not shown). Bottom: 121 realizations of the -1m maneuver under wind uncertainty, presented from the initial condition $x_0 = [0.0, 0.0, 0.0, 0.0]^T$.

by the following differential equations:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v\cos\theta + w_x \\ v\sin\theta + w_y \\ \omega \\ u \end{bmatrix} \tag{12}$$

Here, u is the control input, θ is the current heading, and ω is the angular velocity. The parameters w_x, w_y represent the effects of wind. They are i.i.d. Gaussian random variables with mean 0 and standard deviation 0.2 m/s. This dynamical system is invariant to the action of $SE(2)$, the planar rotation and translation group, which is shown in [29] to be a Lie group. Note that a proof of this invariance under wind uncertainty is straightforward and omitted here for space reasons. A nominal velocity of $v = 10$ m/s is selected, and the control is bounded such that $u \in [-1000 \text{ rad}/s^2, 1000 \text{ rad}/s^2]$. With winds set to 0, this system is identical to the nominal system in [8], but different forms of uncertainty will be imposed in this work. The states x, y, θ are the cyclic coordinates and ω is the only non-cyclic coordinate.

Maneuvers are constructed using direct collocation [30] to generate nominal trajectories satisfying the system dynamics while satisfying control saturation limits. These trajectories are shown in Figure 1 (top, in green). Maneuvers move the vehicle forward 2.25 meters and left or right by up to 1 meter. They are compatible with any state such that $\omega = 0$. Maneuvers will be referred to by the distance they move the vehicle left; the top maneuver in Fig. 1 is +1m. Figure 1 does not show the negative nominal maneuvers, nor the short "recovery" maneuvers used to regulate $\omega \to 0$ if tracking errors cause rotational velocity to accumulate. In the bottom of Fig. 1, the 121 realizations of the -1m maneuver used for planning are shown. In all simulation results in the next section, the nominal trajectories are tracked by a time-varying LQR controller with identity cost matrices.

V. Simulation Results

This section demonstrates performance of the motion primitive planner described above for the example uncertain system. In the following examples, the cost function J being minimized is the distance to the goal at the end of a maneuver.

A. Path Through a Gap

In the first example, the vehicle chooses whether to travel around a wall or through a narrow gap, while its motion is perturbed by a random but uniform wind field. The winds are randomized at each maneuver, but are held constant (at this randomized value) throughout the execution of a maneuver. Winds for one maneuver are conditionally independent of the winds during past or future maneuvers.

In this example, the obstacle is encountered during the first maneuver – therefore, the planning horizon is only a single maneuver. Note, however, that when simulating vehicle motion, a new path is planned after each maneuver is executed. Since the paths to the goal tend to be two maneuvers long, this results in the possibility of different paths being generated in each simulation, since a different realization of the wind occurs. This feedback path planning has a greater effect in the second example shown in the next section.

Figure 2 shows 10 simulation runs with the risk tolerance r set to 1%, 10%, and 15%. In order to achieve a 1% collision probability, the planner elects to go completely around the wall. Note that, due to a limited number of maneuvers, the planner cannot achieve exactly 1% collision probability; rather, it picks the best maneuver with <=1% chance of collision. In this case, the selected maneuver is the +0.7m maneuver, which carries a predicted collision probability of 0%. In 100,000 runs with randomized winds, 5 collisions occur.

The appeal of explicitly chance-constrained planners is that, by changing a single physically meaningful tuning parameter (the risk tolerance), fundamentally different trajectories can be obtained. Increasing the risk tolerance to 10% causes the planner to select the +0.6m maneuver, which passes closer to the wall and has a predicted 2% chance of collision. From 100,000 runs, the actual risk is 2.6%. Increasing the risk tolerance further to 15% causes the planner to navigate through the gap with the -0.1m maneuver. This path is shorter but more dangerous than going around the wall. The predicted risk for this selected primitive is 12% and the actual risk from 100,000 randomized simulations is 11.5%.

The probability of collision during a maneuver can also be evaluated using a Monte Carlo technique, similar to the methods used in in [2] and [12]. Trajectories are generated offline for randomly sampled wind conditions and stored; at the planning step, these trajectories are shifted to start at the current state and checked for collision. Two proposal distributions are considered: "fair sampling" in which values are drawn directly from the wind distributions, and "uniform sampling" in which a uniform proposal distribution is used

TABLE I

ABSOLUTE ERROR IN COLLISION PROBABILITY PREDICTED BY KOOPMAN COMPARED TO THE MEAN ABSOLUTE PREDICTION ERROR ACROSS 20 EXECUTIONS OF VARIOUS MONTE CARLO APPROACHES.

Task	Maneuver	Absolute Error from Koopman	Mean Absolute Error from Fair			Mean Absolute Error from Uniform		
Path Thru Gap	(# samples)	121	121	200	400	121	200	400
	-0.1	1.328%	2.849%	1.876%	1.150%	2.617%	2.325%	1.795%
	+0.6	0.628%	1.076%	1.100%	0.501%	1.000%	0.629%	0.543%
Mobile Obstacles	(# samples)	360	360	2000	10,000	360	2000	10,000
	+0.1	0.212%	0.728%	0.464%	0.206%	1.605%	0.731%	0.358%
	+0.0	0.486%	0.994%	0.418%	0.235%	2.184%	0.585%	0.386%

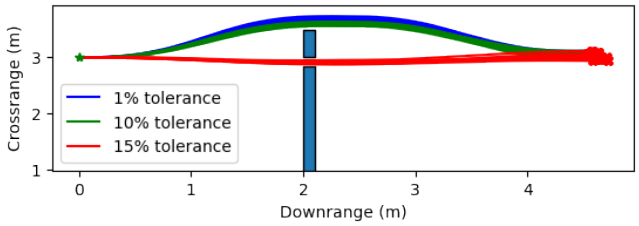

Fig. 2. The vehicle begins at $x_0 = [0.0, 3.0, 0.0, 0.0]^T$ and seeks to navigate to $[4.5, 3.0, 0.0, 0.0]^T$. Ten runs at each tolerance are pictured for illustrative purposes. Red Xs mark locations where a run terminated.

(for further discussion of proposal distributions, see [12] and [13]). The top portion of Table I compares the absolute error in collision probability predicted for the -0.1m and +0.6m maneuvers using the Koopman operator with 121 samples to the average absolute error from 20 executions of fair and uniform Monte Carlo sampling with 121, 200, and 400 samples. All random numbers are generated using the numpy 1.17.4 PCG64 implementation. As shown by the results in Table 1, on average both fair and uniform sampling are less accurate than Koopman with 121 samples. Furthermore, 400 samples are needed to reduce the average Monte Carlo error below that of Koopman with only 121 samples. While the computational effort incurred by running the additional 379 samples required by Monte Carlo to achieve similar accuracy is minimal for this simple system, for systems in which the dynamics are expensive to simulate this difference can be significant.

B. Mobile Obstacles

The second example replaces wind uncertainty with obstacles that move randomly. During each maneuver each obstacle moves in a random direction at a random speed between 0 and 2 m/s. Directions are independently and uniformly distributed and speeds are independently drawn from $N(1,0.2)$. Speeds and directions are constant during the maneuver, and winds are set to 0.

The planner must predict the future locations of the obstacles. The uncertainty space is sampled using 10 points in velocity and 36 points in direction, for a total of 360 samples per obstacle. The planner heuristic assumes that a particular obstacle motion sample continues at the same direction and speed during all primitives in a given path.

Figure 3 shows example planner solutions with three obstacles for $r = 10\%$ and $r = 20\%$. The algorithm replans after each maneuver, so each run can result in a different

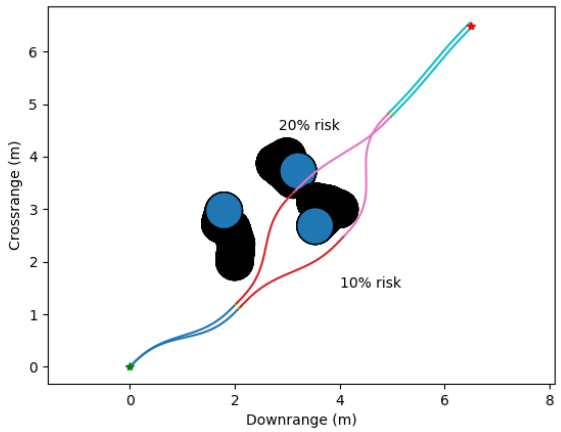

Fig. 3. To achieve a <10% risk of collision, the planner elects to swing wide around the obstacle field. At 20% risk tolerance, it can pass between the obstacles and achieve a shorter path.

path. The blue circles show the obstacle locations at the final time while the black show the locations at each time step (the upper left obstacle interfered with the 20% path only after the vehicle was already past). At $r = 10\%$, the path moves widely around the obstacle field. One thousand runs result in 12 collisions. Increasing the planner's tolerance to 20% causes it to accept narrower clearance with respect to the first obstacle and to pass between the second pair, leading to a shorter overall path. In 1,000 runs at 20% risk, there are 91 collisions. The plotted 10% and 20% runs use the same seed for comparison purposes.

The bottom part of Table I compares collision probability predictions via Koopman and Monte Carlo for two maneuvers available at the initial state of the mobile obstacle task. From one million runs, the true collision probability for the +0.1 maneuver is 5.239%; for the +0.0 maneuver it is 6.913%. On the +0.1 maneuver, fair Monte Carlo needs 10,000 samples to match the accuracy of Koopman with only 360 samples. The +0.0 maneuver is more forgiving; fair Monte Carlo needs only 2,000 samples to outperform Koopman with 360. Uniform sampling is less accurate than fair sampling at all tested sample counts.

VI. CONCLUSION

The algorithm presented in this work offers two key advantages over previous motion primitive-based approaches for systems subject to parametric uncertainty. First, in contrast to both robust and deterministic approaches, the

chance-constrained formulation allows path optimization to be smoothly traded for collision risk. Secondly, compared to Monte Carlo-based schemes, the explicit Koopman operator approach allows the planner to more accurately quantify risk with a lower number of samples, reducing computational effort to achieve a given level of accuracy. While this work represents a promising initial step toward probabilistic motion primitive planning, further work is needed to generalize the formulation to more complicated systems in which the non-cyclic coordinates cannot be directly controlled, and to demonstrate applicability to higher-dimensional systems.

REFERENCES

[1] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1077–1091, Dec 2005.

[2] A. Thakur, P. Svec, and S. K. Gupta, "Gpu based generation of state transition models using simulations for unmanned surface vehicle trajectory planning," *Robotics and Autonomous Systems*, vol. 60, no. 12, pp. 1457 – 1471, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889012001121

[3] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli, "Predictive control for agile semi-autonomous ground vehicles using motion primitives," in *2012 American Control Conference (ACC)*, June 2012, pp. 4239–4244.

[4] I. Saha, R. Ramaithitima, V. Kumar, G. J. Pappas, and S. A. Seshia, "Automated composition of motion primitives for multi-robot systems from safe LTL specifications," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2014, pp. 1525–1532.

[5] A. A. Paranjape, K. C. Meier, X. Shi, S.-J. Chung, and S. Hutchinson, "Motion primitives and 3D path planning for fast flight through a forest," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 357–377, 2015. [Online]. Available: https://doi.org/10.1177/0278364914558017

[6] M. Vukosavljev, Z. Kroeze, M. E. Broucke, and A. P. Schoellig, "A framework for multi-vehicle navigation using feedback-based motion primitives," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 223–229.

[7] T. Schouwenaars, B. Mettler, E. Feron, and J. P. How, "Robust motion planning using a maneuver automaton with built-in uncertainties," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 3, June 2003, pp. 2211–2216 vol.3.

[8] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, June 2017.

[9] A. Charnes, W. W. Cooper, and G. H. Symonds, "Cost horizons and certainty equivalents: An approach to stochastic programming of heating oil," *Management Science*, vol. 4, no. 3, pp. 235–263, 1958. [Online]. Available: https://doi.org/10.1287/mnsc.4.3.235

[10] A. T. Schwarm and M. Nikolaou, "Chance-constrained model predictive control," *AIChE Journal*, vol. 45, no. 8, pp. 1743–1752, 1999.

[11] L. Blackmore, Hui Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *2006 American Control Conference*, June 2006, pp. 7–14.

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

[13] L. Janson, E. Schmerling, and M. Pavone, *Monte Carlo Motion Planning for Robot Trajectory Optimization Under Uncertainty*. Cham: Springer International Publishing, 2018, pp. 343–361. [Online]. Available: https://doi.org/10.1007/978-3-319-60916-4_20

[14] N. E. Du Toit and J. W. Burdick, "Probabilistic Collision Checking With Chance Constraints," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 809–815, Aug 2011.

[15] M. P. Vitus, W. Zhang, and C. J. Tomlin, "A hierarchical method for stochastic motion planning in uncertain environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2263–2268.

[16] A. Mesbah, S. Streif, R. Findeisen, and R. D. Braatz, "Stochastic nonlinear model predictive control with probabilistic constraints," in *2014 American Control Conference*, June 2014, pp. 2413–2419.

[17] M. Ilg, J. Rogers, and M. Costello, "Projectile monte-carlo trajectory analysis using a graphics processing unit," in *AIAA Atmospheric Flight Mechanics Conference*, 2011. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2011-6266

[18] P. Dutta and R. Bhattacharya, "Hypersonic State Estimation Using the Frobenius-Perron Operator," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 325–344, 2011. [Online]. Available: https://doi.org/10.2514/1.52184

[19] A. Halder and R. Bhattacharya, "Dispersion Analysis in Hypersonic Flight During Planetary Entry Using Stochastic Liouville Equation," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 459–474, 2011. [Online]. Available: https://doi.org/10.2514/1.51196

[20] P. Dutta, A. Halder, and R. Bhattacharya, "Uncertainty quantification for stochastic nonlinear systems using Perron-Frobenius operator and Karhunen-Loève expansion," in *2012 IEEE International Conference on Control Applications*, Oct 2012, pp. 1449–1454.

[21] J. J. Meyers, A. M. Leonard, J. D. Rogers, and A. R. Gerlach, "Koopman Operator Approach to Optimal Control Selection Under Uncertainty," in *Proceedings of the 2019 American Control Conference, 2019*, 2019.

[22] A. Leonard, J. Rogers, and A. Gerlach, "Koopman operator approach to airdrop mission planning under uncertainty," *Journal of Guidance, Control, and Dynamics*, Jul 2019. [Online]. Available: https://doi.org/10.2514/1.G004277

[23] I. Mezić and A. Banaszuk, "Comparison of systems with complex behavior," *Physica D: Nonlinear Phenomena*, vol. 197, no. 1, pp. 101 – 133, 2004.

[24] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–driven approximation of the koopman operator: Extending dynamic mode decomposition," *Journal of Nonlinear Science*, vol. 25, no. 6, pp. 1307–1346, Dec 2015.

[25] S. Klus, P. Koltai, and C. Schütte, "On the numerical approximation of the Perron-Frobenius and Koopman operator," *Journal of Computational Dynamics*, vol. 3, no. 1, pp. 51–79, 2016, cited By 31.

[26] Steven L. Brunton and Binbni W. Brunton and Joshua L. Proctor and J. Nathan Kutz, "Koopman Invariant Subspaces and Finite Linear Representations of Nonlinear Dynamical Systems for Control," *PloS One*, vol. 11, no. 2, 2016.

[27] M. Korda and I. Mezic, "Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control," *Automatica*, vol. 93, pp. 149 – 160, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000510981830133X

[28] I. Abraham and T. D. Murphey, "Active Learning of Dynamics for Data-Driven Control Using Koopman Operators," *IEEE Transactions on Robotics*, vol. 35, no. 5, pp. 1071–1083, Oct 2019.

[29] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. CRC Press, Mar 1994.

[30] J. K. Moore and A. van den Bogert. (2017, June) opty: Software for trajectory optimization and parameter identification using direct collocation. Online. [Online]. Available: https://github.com/csu-hmc/opty