SwarmCCO: Probabilistic Reactive Collision Avoidance for Quadrotor Swarms Under Uncertainty

Senthil Hariharan Arul and Dinesh Manocha

***Abstract*—We present decentralized collision avoidance algorithms for quadrotor swarms operating under uncertain state estimation. Our approach exploits the differential flatness property and feedforward linearization to approximate the quadrotor dynamics and performs reciprocal collision avoidance. We account for the uncertainty in position and velocity by formulating the collision constraints as chance constraints, which describe a set of velocities that avoid collisions with a specified confidence level. We present two different methods for formulating and solving the chance constraints: our first method assumes a Gaussian noise distribution. Our second method is its extension to the non-Gaussian case by using a Gaussian Mixture Model (GMM). We reformulate the linear chance constraints into equivalent deterministic constraints, which are used with an MPC framework to compute a local collision-free trajectory for each quadrotor. We evaluate the proposed algorithm in simulations on benchmark scenarios and highlight its benefits over prior methods. We observe that both the Gaussian and non-Gaussian methods provide improved collision avoidance performance over the deterministic method. On average, the Gaussian method requires $\sim$5 ms to compute a local collision-free trajectory, while our non-Gaussian method is computationally more expensive and requires $\sim$9 ms on average in scenarios with 4 agents.**

***Index Terms*—Collision avoidance, motion and path planning, simulation and animation.**

I. Introduction

Recent advances in Unmanned Aerial Vehicles (UAVs) have led to many new applications for aerial vehicles. These include search and rescue, last-mile delivery, and surveillance, and they benefit from the small size and maneuverability of quadrotors. Furthermore, many of these applications use a large number of quadrotors (e.g., swarms). A key issue is developing robust navigation algorithms so that each quadrotor agent avoids collisions with other dynamic and static obstacles in its environment. Moreover, in general, the quadrotors need to operate in uncontrolled outdoor settings like urban regions, where the agents rely on onboard sensors for state estimation. In practice, onboard sensing can be noisy, which can significantly affect the collision avoidance performance.

Manuscript received September 30, 2020; accepted February 6, 2021. Date of publication February 24, 2021; date of current version March 15, 2021. This letter was recommended for publication by Associate Editor M. Morales and Editor N. Amato upon evaluation of the reviewers' comments. This work was supported in part by ARO under Grants W911NF1810313 and W911NF1910315 and in part by Intel. *(Corresponding author: Senthil Hariharan Arul.)*

Senthil Hariharan Arul is with the Department of Electrical, and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: sarul1@umd.edu).

Dinesh Manocha is with the Department of Computer Science, University of Maryland, College Park, MD 20742 USA (e-mail: dmanocha@gmail.com).

Digital Object Identifier 10.1109/LRA.2021.3061975

Prior work on collision-free navigation is broadly classified into centralized and decentralized methods. Centralized methods [1]–[3] plan collision-free trajectories for all agents in a swarm simultaneously, and they can also provide guarantees on smoothness, time optimality, and collision avoidance. However, due to the centralized computation, these algorithms do not scale well with the number of agents. In decentralized methods [4]–[8], each agent makes independent decisions to avoid a collision. In practice, they are scalable due to the decentralized decision making, but do not guarantee optimality or reliably handle uncertainty.

In addition, prior work on multi-agent collision avoidance is limited to deterministic settings. These methods are mainly designed for indoor environments, where the physical evaluations are performed with a MoCap-based state estimation. On the other hand, real-world quadrotor deployment relies on onboard sensor data, which can be noisier. For example, depth cameras are widely used in robotics applications, but the estimated depth values may have errors due to lighting, calibration, or object surfaces [9], [10]. Some of the simplest techniques consider zero-mean Gaussian uncertainty by enlarging the agent's bounding geometry based on the variance of uncertainty [11], [12]. However, these methods tend to over-approximate the collision probability, resulting in conservative navigation schemes [13]. Other uncertainty algorithms are based on chance-constraint methods [13], [14]. These algorithms are less conservative in practice, but assume simple agent dynamics or are limited to simple scenarios.

A. Main Results:

We present a decentralized, probabilistic collision avoidance method (SwarmCCO) for quadrotor swarms operating in dynamic environments. Our approach builds on prior techniques for multi-agent navigation based on reciprocal collision avoidance [6], [15], and we present efficient techniques to perform probabilistic collision avoidance by chance-constrained optimization (CCO). We handle the non-linear quadrotor dynamics using flatness-based feedforward linearization. The reciprocal collision avoidance constraints are formulated as chance constraints and combined with the MPC (Model Predictive Control) framework.

- Our first algorithm assumes a Gaussian noise distribution for the state uncertainties and reformulates the collision avoidance chance constraints as a set of deterministic second-order cone constraints.

- Our second algorithm is designed for non-Gaussian noises. We use a Gaussian Mixture Model (GMM) to approximate the noise distribution and replace each collision avoidance constraint using a second-order cone for each Gaussian components. The cone constraints for the individual Gaussian components are related to the GMM's probability distribution by using an additional constraint based on GMM's mixing coefficient.

We evaluate our probabilistic methods (SwarmCCO) in simulated environments with a large number of quadrotor agents. We compare our probabilistic method's performance with the deterministic algorithm [15] in terms of the path length, time to goal, and the number of collisions. We observe that both our Gaussian and non-Gaussian methods result in fewer collisions in the presence of noise. Our average computation time is ~ 5 ms per agent for our Gaussian method and ~ 9 ms per agent for the non-Gaussian method in scenarios with 4 agents. The non-Gaussian method is computationally expensive compared to the Gaussian method, but the non-Gaussian method provides improved performance in terms of shorter path lengths, and satisfying collision avoidance constraints (Section V-D). Hence, the non-Gaussian method tends to offer better performance in constricted regions due to better approximation of noise, where the Gaussian method may result in an infeasible solution.

The letter is organized as follows. Section II summarizes the recent relevant works in probabilistic collision avoidance. Section III provides a brief introduction to DCAD [15] and ORCA [6] chance-constraints. In Section IV, we present our algorithms and describe the chance constraint formulation. In Section V, we present our results and compare the performance with other methods.

II. Previous Work

In this section, we provide a summary of the recent work on collision avoidance and trajectory planning under uncertainty.

A. Decentralized Collision Avoidance With Dynamics

Decentralized collision avoidance methods [4]–[7], [16], [17] avoid collisions by locally altering the agent's path based on the local sensing information and state estimation. Velocity Obstacle (VO) [4] methods such as RVO [5] and ORCA [6] provide decentralized collision avoidance for velocity-controlled agents. This concept was extended to acceleration-controlled agents in the AVO algorithm [16] and used to generate n^{th} order continuous trajectories in [18]. Morgan *et al.* [7] described a sequential convex programming (SCP) method for trajectory generation. Most of these methods have been designed for deterministic settings, where exact state information is available. Under imperfect state estimation and noisy actuation, the performance of deterministic algorithms may not be reliable and can lead to collisions [14], [19]. Hence, we need probabilistic collision avoidance methods for handling uncertainty.

TABLE I
Notation and Symbols

Notation	Definition
$\mathbf{r}_i$	3-D position of i^{th} quadrotor given by $[r_{i,x}, r_{i,y}, r_{i,z}]$
$\mathbf{v}_i, \mathbf{a}_i$	3-D Velocity and Acceleration of i^{th} quadrotor given by $[v_{i,x}, v_{i,y}, v_{i,z}]$ and $[a_{i,x}, a_{i,y}, a_{i,z}]$, respectively
R_i	Radius of agent i's enclosing sphere
ϕ, θ, ψ	Roll, pitch and yaw of the quadrotor, respectively
$\mathbf{v}_i^{orca}$	Collision avoiding velocity for agent i satisfying ORCA constraint.
$f^{orca_i^j}$	ORCA plane constraint between agents i and j given by $\mathbf{m}^T\mathbf{v}_i^{orca} - b \geq 0$. $\mathbf{m}$ and b are functions of the agents trajectory.
μ_z, σ_z	Mean and standard deviation of some variable 'z'
$\boldsymbol{\mu}_\mathbf{z}, \Sigma_\mathbf{z}$	Mean and covariance of some vector '$\mathbf{z}$'
$P(x)$	Probability of x
$\mathbf{x}, \mathbf{u}$	Quadrotor flat state and flat control input

B. Uncertainty Modeling

Snape *et al.* [11] extended the concept of VO to address state estimation uncertainties using Kalman filtering and bounding volume expansion for single-integrator systems. That is, the agent's bounding polygon is enlarged based on the co-variance of uncertainty. Kamel *et al.* [12] proposed an nonlinear MPC formulation for quadrotor collision avoidance and used the bounding volume expansion to address sensor uncertainties. DCAD [15] presented a collision avoidance method for quadrotors using ORCA and bounding volume expansion. Bounding volume expansion methods retain the linearity of ORCA constraints; hence, they are fast but tend to be conservative. In practice, they can lead to infeasible solutions in dense scenarios [13]. Angeris *et al.* [20] accounted for uncertainty in estimating a neighbor's position using an ellipsoid-based formulation before computing a safe reachable set for the agent [21].

In contrast to bounding volume methods, [13], [22] modeled the stochastic collision avoidance as a chance-constrained optimization problem. These techniques assumed a Gaussian noise distribution for the position and transformed the chance constraints to deterministic constraints using the mean and covariance of the Gaussian distribution. Gopalakrishnan *et al.* [14] presented PRVO, a probabilistic version of RVO for Gaussian noise distribution by approximating the chance constraints using Cantelli's bound. However, PRVO considers simple single-integrator dynamics. Jyotish *et al.* [23] extended PRVO to non-parametric noise, formulated the CCO problem as a distribution matching problem, and solved it using RKHS (Reproducing Kernel Hilbert Space) embedding for a linear system. However, this method is computationally expensive and requires ~ 200 ms to compute a suitable velocity in the presence of 2 neighbors.

III. Background and Problem Formulation

This section discusses the problem statement and gives an overview of various concepts used in our approach. Table I summarizes the symbols and notations used in the letter.

A. Problem Statement

We consider N agents occupying a workspace $\mathcal{W} \subseteq \mathbb{R}^3$. Each agent $i \in \{1, 2, \ldots, \mathrm{N}\}$ is modeled with non-linear quadrotor dynamics, as described in [15]. For simplicity, each agent's geometric representation is approximated as a sphere of radius R. We assume that each agent knows its neighbor's position and velocity either through inter-agent communication or visual sensors, and this information is not precise. No assumption is made on the nature of the uncertainty distribution in this information; hence, the random variables are assumed to be non-parametric (i.e. they are assumed to follow no particular family of probability distribution). Our algorithm approximates the distribution using a Gaussian Mixture Model GMM.

At any time instant, two agents i and j, where $i, j \in \{1, 2, \ldots, \mathrm{N}\}$, $i \neq j$, are said to be collision-free if their separation is greater than sum of their bounding sphere radii. That is, $\|\mathbf{r}_i - \mathbf{r}_j\|_2 \geq R_i + R_j$. Since the position and velocity are random variables, collision avoidance is handled using a stochastic method based on chance constraints.

B. Orca

ORCA [6] is a velocity obstacle-based method that computes a set of collision-free velocities. Each ORCA constraint being linear represents a half-space of collision-free velocities which can be represented as follows:

$$f^{ORCA^i_j}(.) = \mathbf{m}^T \mathbf{v}_i^{orca} - b \geq 0. \tag{1}$$

Here, $\mathbf{v}_i^{orca}$ is any velocity in the half-space of collision-free velocities. Variables $\mathbf{m}$ and b are functions of the agent trajectories. More details on this are available in [24] and [14]. Eqn. 1 is used to compute the chance constraint (see Section IV).

C. Differential Flatness and Feedforward Linearization

A nonlinear system $\dot{\mathbf{x}}_o = f(\mathbf{x}_o, \mathbf{u}_o)$ is differentially flat if a set $\zeta = [\zeta_1, \zeta_2, \ldots, \zeta_m]$ of differentially independent components and their derivatives can be used to construct the system state space and control inputs [15]. Given a nonlinear, differentially flat system and a smooth reference trajectory in ζ (denoted ζ_{ref}), the nonlinear dynamics can be represented as a linear flat model ($\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}$) in terms of the flat state ($\mathbf{x}$) and input ($\mathbf{u}$). Provided a control input $\mathbf{u}_o^*$ computed from ζ_{ref} (and its derivative) is applied to the nonlinear system, and the initial flat state $\mathbf{x}(0)$ equals the initial condition of the reference state trajectory $\mathbf{x}_{ref}(0)$ (computed from ζ_{ref} and its derivatives). A quadrotor is a differential flat system for the choice of differentially independent set $\zeta = [r_x, r_y, r_z, \psi]$ [25]. We use the linear flat model in the MPC problem (as shown in Equation 2), and compute $\mathbf{x}_{ref}$ and $\mathbf{u}$, which are used to compute a control input $\mathbf{u}_o^*$ for the quadrotor using an inverse mapping. Please refer to our ArXiv report [24] for more details on flatness.

D. Dcad

Decentralized Collision Avoidance with Dynamics (DCAD) [15] exploits the differential-flatness property of a quadrotor to feedforward linearize the quadrotor dynamics and uses linear MPC along with ORCA constraints to plan a collision-free trajectory. In addition, DCAD uses an inverse mapping to compute the control input ($\mathbf{u}_o$) for the nonlinear quadrotor system from the flat states ($\mathbf{x}$) and inputs ($\mathbf{u}$). Our method differs from DCAD as we pose the ORCA linear constraints as chance constraints and use a chance-constrained optimization algorithm to compute a collision-avoiding input for the quadrotor. Additionally, we consider the flat state and inputs to be given by $\mathbf{x} = [\mathbf{r}_i, \mathbf{v}_i, \psi]$. and $\mathbf{u} = [\mathbf{a}_i, \dot{\psi}]$ similar to [26]. Please refer to our ArXiv report [24] for more details on the use of inverse mapping.

E. Chance Constraints

Chance-constrained optimization is a technique for solving optimization problems with uncertain variables [27], [28]. A general formulation for chance-constrained optimization is given as

$$\begin{aligned} &\text{minimize} && \gamma \\ &\text{subject to} && \mathrm{P}(f(x) \leq \epsilon) \geq \delta. \end{aligned}$$

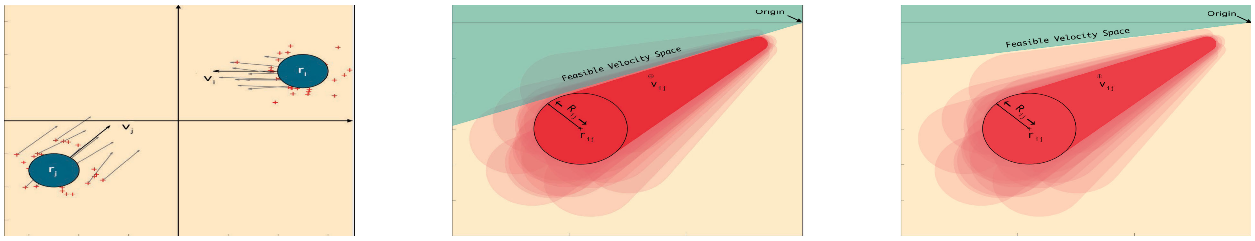
Here, γ is the objective function for the optimization. $f(x)$ is the constraint on the random variable x. $f(x)$ is said to be satisfied when $f(x) \leq \epsilon$. Since x is a random variable, the constraint $f(x) \leq \epsilon$ is formulated as the probability of satisfying of the constraint. That is, $\mathrm{P}(f(x) \leq \epsilon) \geq \delta$ is the chance constraint and is said to be satisfied when the probability of satisfying the constraint $f(x) \leq \epsilon$ is over a specified confidence level, δ.

IV. SwarmCOO: Probabilistic Multi-Agent Collision Avoidance

In this section, we describe our MPC optimization problem and summarize our Gaussian and non-Gaussian chance constraint formulations for collision avoidance. Fig. 1 highlights a 2D scenario for collision avoidance between two agents with state uncertainty. The figure shows a distribution of Velocity Obstacle (VO) cones constructed for the given position and velocity distribution. We observe that the collision-free relative velocity set computed using deterministic ORCA overlaps with a portion of this VO distribution. Hence, a velocity chosen from this set can result in a collision. In contrast, our formulation based on chance constraints results in a feasible relative velocity set that has a higher probability of being collision-free.

A. MPC Optimization Setup

We use a receding horizon planner to generate the collision-free trajectories for each quadrotor agent. The underlying optimization framework is common to both our Gaussian and non-Gaussian SwarmCCO formulations. Our Gaussian and non-Gaussian methods differ in the formulation for collision avoidance chance constraints, i.e. the constraint $\mathrm{P}(\mathbf{m}^T \mathbf{v}_i^{orca} - b \leq 0) \leq \delta$ in the optimization is formulated differently (IV-C). Each pair of agents has a constraint $\mathrm{P}(\mathbf{m}^T \mathbf{v}_i^{orca} - b \leq 0) \leq \delta$. That is, if an agent has 5 neighbors, there would be 5 chance constraints in Eqn. (2). The variables $\mathbf{m}$ and b for each constraint would depend on the relative positions and velocities between that pair of agents. For clarity, we have considers a single neighbor case in this section. Each quadrotor agent computes the collision avoidance constraints



(a) Two-robot scenario (b) Deterministic ORCA (c) Chance Constrained ORCA

Fig. 1. Deterministic and Probabilistic collision avoidance between two agents. (a) Two circular agents with mean positions r_i and r_j and their respective mean velocities v_i and v_j (indicated by the black arrow) are shown. The red '+' markers indicate position samples from the position's uncertainty distribution, and the grey arrows indicate velocity samples from the velocity's uncertainty distribution. (b) The *darker* red cone represents the Velocity Obstacle (VO) constructed using the mean position and velocity, while the *lighter* or *translucent* red cones represent the VOs constructed from the position and velocity samples from the uncertainty distribution. The green region is the feasible region for the relative velocity computed using ORCA. As can be seen, the green regions overlap with parts of VOs constructed using the position and velocity samples, which can lead to collisions. (c) Feasible relative velocity set computed using the chance constraint ORCA method avoids a major portion of the VO samples.

at each time step and plans a trajectory for the next N time steps. The trajectory is re-planned continuously to account for the changes in the environment.

In our optimization formulation (given below), "N" represents the prediction horizon. The weight matrices, Q and R, prioritize between trajectory tracking error and the control input, respectively. $\mathbf{x}_k$ represents the state of the agent at time step k, and the matrix A and B are the system matrices for the linearized quadrotor model. Velocity and acceleration are constrained to maximum values, $v_{\max}$ and $a_{\max}$, respectively, and are realized using box constraints.

$$\begin{aligned} \text{minimize} \quad & \sum_{t=0}^{N} (\mathbf{x}_{ref,t} - \mathbf{x}_t)\, Q\, (\mathbf{x}_{ref,t} - \mathbf{x}_t) + \mathbf{u}_t R \mathbf{u}_t \\ \text{subject to} \quad & \mathbf{x}_0 = \mathbf{x}_t, \\ & \mathbf{x}_{k+1} = A\mathbf{x}_k + B\mathbf{u}_k, \\ & ||\mathbf{v}_k|| \leq v_{\max},\ ||\mathbf{a}_k|| \leq a_{\max} \\ & \mathrm{P}\left(\mathbf{m}^T \mathbf{v}_i^{orca} - b \geq 0\right) \geq \delta, \\ & \forall k = 0, 1, \ldots, N-1. \end{aligned} \tag{2}$$

Our MPC-based algorithm plans in terms of acceleration and yaw rate, which constitute the control input (Section III-D). The constraint $\mathrm{P}(\mathbf{m}^T \mathbf{v}_i^{orca} - b \leq 0) \leq \delta$ represents the chance constraints defined on ORCA, i.e. the constraint is said to be satisfied if, given the uncertainty in state, the probability that the ORCA constraint is satisfied is greater than δ. The output of the MPC is the control input vector for the quadrotor.

B. Collision Avoidance Velocity

The VO is constructed using the relative position $(\mathbf{r}_i - \mathbf{r}_j)$ and velocity $(\mathbf{v}_A - \mathbf{v}_B)$ of the agents. From Fig. 1(b), we notice that the ORCA plane passes through the origin. Thus, the parameter b in the constraint (1) is zero in this case. The feasible velocity set for agent A is constructed by translating this plane by the agent's mean velocity plus 0.5 times (as in [6]) the change in relative velocity proposed by ORCA. Due to this translation, b can be non-zero, and we use a mean value for b instead of a distribution to apply the formulation below.

C. Chance Constraint Formulation

Since the position and velocity data of the agent and its neighbors are non-Gaussian random variables, the collision avoidance constraints must consider the uncertainty in the agent's state estimations for safety. As mentioned in Section III-A, we do not make any assumption on the nature of the uncertainty distribution. However, we model uncertainty using two different methods. Our first method approximates the noise distribution as a Gaussian distribution, which works well for certain sensors. In comparison, our second method is more general and works with non-Gaussian uncertainty by fitting a Gaussian Mixture Model to the uncertainty distribution.

Method I: Gaussian Distribution

In this method, we approximate the position and velocity uncertainties using a multivariate Gaussian distribution. For an agent i, its position and velocity variables are approximated as $\mathbf{r}_i \sim N(\boldsymbol{\mu}_{i,r}, \Sigma_{i,r})$, and $\mathbf{v}_i \sim N(\boldsymbol{\mu}_{i,v}, \Sigma_{i,v})$. The deterministic ORCA constraint between two agents i and j is given by the following plane (linear) equation:

$$f^{ORCA_j^i}(.) = \mathbf{m}^T \mathbf{v}_i^{orca} - b. \tag{3}$$

Here, the parameters $\mathbf{m}$ and b are functions of the agent's trajectory. In a stochastic setting, the parameters $\mathbf{m}$ and b are random variables due to their dependence on the agent's position and velocity. Though the uncertainty in position and velocity are assumed to be Gaussian for this case, this need not translate to a Gaussian distribution for $\mathbf{m}$. However, we approximate $\mathbf{m}$'s distribution as Gaussian for the application of our algorithm, with expectation and covariance represented by $\boldsymbol{\mu}_{\mathbf{m}}$ and $\Sigma_{\mathbf{m}}$, respectively. Eqn. (4) highlights the chance constraint, representing the probability that the ORCA constraint (3) is satisfied, given the uncertainties in the position and velocity data. This probability is set to be above a predefined confidence level (δ).

$$P(\mathbf{m}^T \mathbf{v}_i^{orca} - b \geq 0) \geq \delta, \ \ \mathbf{m} \sim N(\boldsymbol{\mu}_{\mathbf{m}}, \Sigma_{\mathbf{m}}). \tag{4}$$

From [28], we know that if $\mathbf{a}$ follows a Gaussian distribution, the chance constraint can be transformed into a deterministic second-order cone constraint. This is summarized in Lemma IV.1.

Lemma IV.1: For a multivariate random variable $\mathbf{m} \sim N(\boldsymbol{\mu}_\mathbf{m}, \Sigma_\mathbf{m})$, the chance constraint $P(\mathbf{m}^T\mathbf{x_t} \leq b) > \delta$ can be reformulated as a deterministic constraint on the mean and covariance.

$$P(\mathbf{m}^T\mathbf{x}_t \leq b) > \delta \iff b - \boldsymbol{\mu}_\mathbf{m}^T\mathbf{x}_t \geq \text{erf}^{-1}(\delta) \left\| \Sigma_\mathbf{m}^{\frac{1}{2}}\mathbf{x}_t \right\|_2, \tag{5}$$

where erf(x) is the standard error function given by, $\text{erf}(x) = \frac{1}{2\pi}\int_0^x e^{-\tau^2/2}d\tau$.

Here, δ is the confidence level associated with satisfying the constraint $\mathbf{m}^T\mathbf{x_t} \leq b$. Since our collision avoidance constraints are linear, each chance constraint can be reformulated to a second order cone constraint, as shown in Lemma IV.1. Hence, each collision avoidance chance constraint can be written as

$$P(\mathbf{m}^T\mathbf{v}_i^{orca} - b \geq 0) \geq \delta \iff$$
$$\boldsymbol{\mu}_\mathbf{m}^T\mathbf{v}_i^{orca} - b + \text{erf}^{-1}(\delta) \left\| \Sigma_\mathbf{m}^{\frac{1}{2}}\mathbf{v}_i^{orca} \right\|_2 \leq 0. \tag{6}$$

Method II: Gaussian Mixture Model (GMM)

To handle non-Gaussian uncertainty distributions, we present an extension of the Gaussian formulation (Method I). In this case, the probability distribution for the position and velocity is assumed to be non-parametric and non-Gaussian, i.e. the probability distribution is not known. We assume that we have access to s samples of these states that could come from a black-box simulator or a particle filter. Using s samples, a distribution for the parameter $\mathbf{m}$ is constructed. In our implementation, we use a value of $s = 40$ samples.

A GMM model of n Gaussian components is fit to the probability distribution of parameter $\mathbf{m}$ in Eqn. 3 using Expectation-Maximization (EM). Each collision avoidance constraint is split into n second-order constraints, each corresponding to a single Gaussian component. Furthermore, an additional constraint is used that relates the $\mathbf{n}$ second-order constraints to GMM probability distribution. From [29], we know that if $\mathbf{m}$ follows a GMM distribution, the chance constraint can be transformed into a set of deterministic constraints. This is summarized in Lemma IV.2.

Lemma IV.2: For a non-Gaussian random variable $\mathbf{m}$ and a linear equation $f(\mathbf{x_t}) = \mathbf{m}^T\mathbf{x_t} \leq b$, the chance constraint $P(\mathbf{m}^T\mathbf{x_t} \leq b) > \delta$ can be reformulated as a set of deterministic constraints on the mean and co-variance. Let the distribution of $\mathbf{m}$ be approximated by $\mathbf{n}$ Gaussian components. The probability that $f(\mathbf{x_t})$ is satisfied while $\mathbf{m}$'s distribution is given by the n^{th} Gaussian component of the GMM is given by

$$P_i = \text{erf}\left(\frac{b - \boldsymbol{\mu}_\mathbf{m}^T\mathbf{x}_t}{\sqrt{\mathbf{x_t}^T\Sigma_\mathbf{m}\mathbf{x_t}}}\right). \tag{7}$$

Then, the probability that $f(\mathbf{x_t})$ is satisfied for GMM distribution of $\mathbf{m}$ is given by

$$P_{GMM} = \sum_{i=1}^{n} \alpha_i P_i. \tag{8}$$

Here, α_is are the mixing coefficients for the GMM, satisfying $\sum_{i=1}^n \alpha_i = 1$.

Let us assume that the probability of satisfying $\mathbf{m}^T\mathbf{v}_i^{orca} - b \geq 0$, while considering only the n^{th} Gaussian component, is given by η_i. We can reformulate the constraint using Lemma IV.2, which is given by Eqn. 9.

The probability of satisfying the linear constraint considering the GMM distribution for $\mathbf{m}$ is computed by mixing the individual component probabilities using the mixing coefficients (α_i). A probability δ is chosen as the required confidence level. Eqn. (10) represents the chance constraint that the probability of satisfying Eqn. (3) is greater than δ. The chance constraint can be reformulated as:

$$P(\mathbf{m}^T\mathbf{v}_i^{orca} - b \geq 0) \geq \delta \iff$$
$$\begin{cases} \boldsymbol{\mu}_\mathbf{m}^T\mathbf{v}_i^{orca} - b + \text{erf}^{-1}(\eta_i)\left\|\Sigma_\mathbf{m}^{1/2}\mathbf{v}_i^{orca}\right\|_2 \\ \leq 0, \forall i \in 1 \text{ to n} & (9) \\ \sum_{i=1}^n \alpha_i\eta_i > \delta & (10). \end{cases}$$

In Eqn. 10 the values for the mixing coefficient and confidence are known prior to the optimization. We notice that for a given set of mixing coefficients (α_is) and confidence (δ), multiple sets of values for η_i's can satisfy Eqn. 10. The value of η_i in turn affects the feasible velocity set. Hence, we plan for η_i's in addition to the control input in problem (Equation 2). When GMM has 3 components, i.e. $n = 3$, we have three additional variables given by η_1, η_2, η_3 in the optimization problem. Now the optimization problem (Equation 2) simultaneously computes values for acceleration, $\dot{\psi}$ and η_i, such that the collision avoidance chance constraints (Eqn. 10) are satisfied.

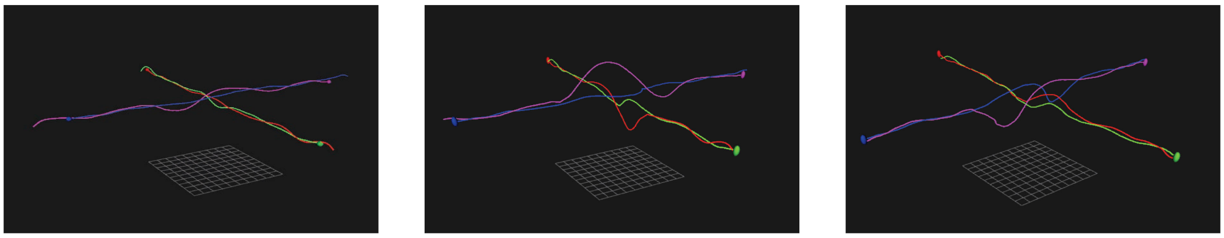
The computed control input from the optimization problem (2) is transformed into an inner-loop control input for the quadrotor using an inverse map, similar to [15].

V. Results

In this section, we describe the implementation of our algorithm and our simulation setup. Further, we summarize our evaluations and highlight the benefits of our approach. Additional comparisons between non-Gaussian SwarmCCO and a conservative method using bounding volume expansion is available in our Arxiv report [24].

A. Experimental Setup

Our method is implemented on an Inter Xeon w-2123 3.6 GHz with 32 GB RAM and a GeForce GTX 1080 GPU. Our simulations are built using the PX4 Software In The Loop (SITL) framework, ROS Kinetic, and Gazebo 7.14.0. We solve the MPC optimization using the IPOPT Library with a planning horizon of $N = 8$ steps and a time step of δt $=$ 100 ms. We consider a non-Gaussian distribution for the position and velocity data, from which the input sensor readings for both the Gaussian and non-Gaussian SwarmCCO methods are generated. Gazebo state information represents the ground truth position and velocity data, while we add non-zero mean, non-Gaussian noise to simulate state uncertainties. The added noise is generated through a GMM model of 3 Gaussian



(a) DCAD(Deterministic) (b) Gaussian Noise Approximation (c) Non-Gaussian Noise Approximation

Fig. 2. The figure illustrates a scenario with 4 agents exchanging positions with their diagonally opposite agents. The collision avoidance is performed based on deterministic DCAD and Gaussian and non-Gaussian SwarmCCO. DCAD results in more collisions because uncertainty is not explicitly considered. We observe that some agents travel a longer path when using the Gaussian formulation than when using the non-Gaussian formulation (Section V-D1).

TABLE II
THE NUMBER OF TRIALS (OUT OF 100) WITH QUADROTOR COLLISIONS WHEN AGENTS TRAVEL TO THEIR ANTIPODAL POSITIONS. WE OBSERVE THAT OUR ALGORITHMS RESULT IN FEWER COLLISIONS, AS COMPARED TO DETERMINISTIC DCAD. WE CONSIDER TWO CASES FOR THE NON-GAUSSIAN SWARMCCO, THE FIRST CASE WITH TWO GAUSSIAN COMPONENTS (N = 2), AND THE OTHER WITH 3 COMPONENTS (N = 3)

Noise	Confidence level	Method	Number of collisions				
			No of Agents				
			2	4	6	8	10
Σ_1		DCAD (deterministic)	27	25	35	40	44
	$\delta = 0.75$	Gaussian SwarmCCO	0	4	4	7	15
		Non-Gaussian SwarmCCO (n=2)	0	2	5	4	12
		Non-Gaussian SwarmCCO (n=3)	0	0	2	5	13
	$\delta = 0.90$	Gaussian SwarmCCO	0	3	2	7	13
		Non-Gaussian SwarmCCO (n=2)	0	1	3	4	12
		Non-Gaussian SwarmCCO (n=3)	0	0	1	2	10
Σ_2		DCAD (deterministic)	56	26	32	51	58
	$\delta = 0.75$	Gaussian SwarmCCO	0	2	3	11	18
		Non-Gaussian SwarmCCO (n=2)	0	1	3	5	9
		Non-Gaussian SwarmCCO (n=3)	0	0	0	3	10
	$\delta = 0.90$	Gaussian SwarmCCO	0	2	1	4	6
		Non-Gaussian SwarmCCO (n=2)	0	0	1	3	4
		Non-Gaussian SwarmCCO (n=3)	0	0	0	2	4

components. We use two different GMM to simulate noise for position readings: $GMM_1 : \mu_{GMM1} = [0.15, 0.08, -0.05]$, $\Sigma_{GMM1} = diag([0.06, 0.7, 0.3])$, and $GMM_2 : \mu_{GMM2} = [0.2, 0.0, -0.2]$, $\Sigma_{GMM2} = diag([1.0, 0.3, 1.0])$. The velocity readings are simulated using a noise distribution that has half the mean and covariance values of Σ_{GMM1} and Σ_{GMM2}. Further, for our evaluation we consider two confidence levels given by $\delta_1 = 0.75$ and $\delta_2 = 0.90$. The RVO-3D library is utilized to compute the ORCA collision avoidance constraints. We consider a sensing region of 8 m for the ORCA plane computation. The agent's physical radius is assumed to be 0.25 m while the agent's radius for the ORCA computation is set as 0.5 m. In our evaluations, two agents are considered to be in collision if their positions are less than 0.5 m apart. Further, we show results for the non-Gaussian method with 2 components ($n = 2$) GMM and 3 components ($n = 3$) GMM.

B. Generated Trajectories

We evaluate our method in simulation with four quadrotors exchanging positions with the antipodal agents (circular scenario). Fig. 2 shows the resulting trajectories for this scenario using deterministic DCAD [15] and Gaussian and non-Gaussian SwarmCCO. We observe that in deterministic DCAD, the agents do not handle noise and hence the trajectories often result in collisions. In Fig. 2 , the DCAD trajectories are shown and consist of multiple collisions. In contrast, the trajectories generated by SwarmCCO methods are safer.

C. Collision Avoidance

We evaluate our method in a circular scenario. Table II summarizes the number of trials with observed collisions out of a total of 100 trials. We observe that the performance of the deterministic method degrades (in terms of number of collisions) with added noise. In contrast, we observe good performance with both the Gaussian and non-Gaussian SwarmCCO. As expected, the number of collisions reduces with an increase in confidence level (δ).

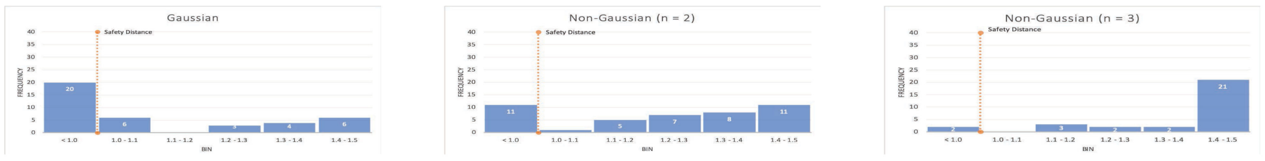
D. Gaussian Vs. Non-Gaussian

In this subsection, we compare the performance of our Gaussian and non-Gaussian formulations for SwarmCCO. We consider the circular scenario where the agents move to their antipodal positions.

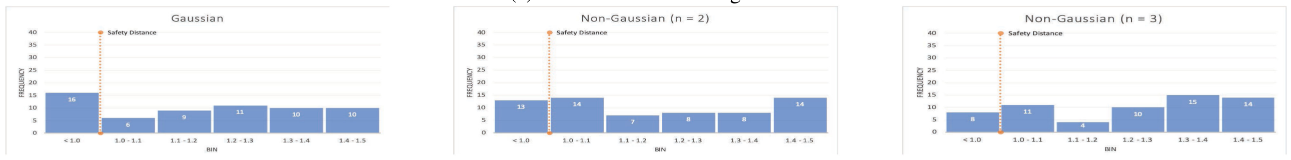
1) Path Length: For each agent, the reference path to its goal is a straight-line path of 40 m length directed to the diametrically opposite position. In Table III, we tabulate the mean path length

TABLE III
AVERAGE PATH LENGTH TRAVELED BY THE AGENT WHILE EXCHANGING POSITIONS WITH THE ANTIPODAL AGENTS. THE REFERENCE, STRAIGHT LINE PATH TO THE GOAL IS 40 M LONG. THE GAUSSIAN METHOD IS RELATIVELY CONSERVATIVE, RESULTING IN LONGER PATH LENGTHS ON AVERAGE COMPARED TO THE NON-GAUSSIAN METHOD. WE OBSERVE THIS PERFORMANCE IN SCENARIOS WITH 8 AND 10 AGENTS

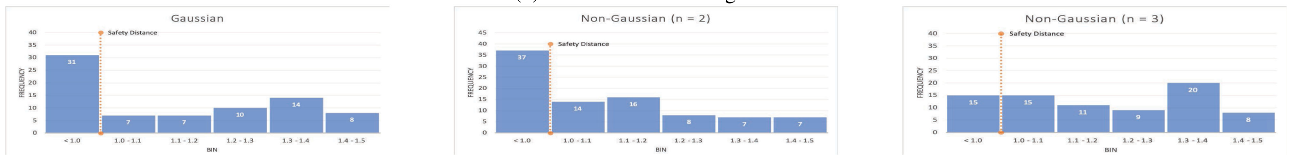
Noise	Confidence Level	Method	Path Length (m)				
			No of Agents				
			2	4	6	8	10
Σ_1	$\delta = 0.75$	Gaussian SwarmCOO	41.12	42.75	44.04	44.88	47.92
		Non-Gaussian SwarmCOO (n=2)	41.06	41.95	43.03	44.39	46.66
		Non-Gaussian SwarmCOO (n=3)	41.06	42.09	43.12	44.59	46.52
	$\delta = 0.90$	Gaussian SwarmCOO	41.10	43.14	45.05	46.64	48.87
		Non-Gaussian SwarmCOO (n=2)	41.07	42.03	43.31	44.42	46.53
		Non-Gaussian SwarmCOO (n=3)	41.06	42.12	43.35	44.57	46.62
Σ_2	$\delta = 0.75$	Gaussian SwarmCOO	41.21	43.06	44.51	46.38	49.61
		Non-Gaussian SwarmCOO (n=2)	41.21	42.67	44.09	45.80	47.87
		Non-Gaussian SwarmCOO (n=3)	41.22	42.47	44.14	45.93	48.15
	$\delta = 0.90$	Gaussian SwarmCOO	41.23	43.69	45.37	47.90	50.41
		Non-Gaussian SwarmCOO (n=2)	41.21	42.47	44.13	45.68	48.09
		Non-Gaussian SwarmCOO (n=3)	41.23	42.70	44.62	46.13	48.25



(a) Scenario with four agents.



(b) Scenario with six agents.



(c) Scenario with eight agents.

Fig. 3. Histogram of least inter-agent distance in the circular scenario with 4, 6 and 8 agents. The agents have a radius of 0.25 m, and the ORCA planes are constructed with an augmented agent radius of 0.5 m. Hence, an inter-agent distance of less than 1 m is a collision according to the ORCA constraint, though the agents do not actually collide. The histogram is constructed over 100 trials, and the trials with least inter-agent distance greater than 1.5 m are not included in the histogram. The safe inter-agent distance of 1 m is denoted by the dotted orange line. We observe that the non-Gaussian method results in fewer trials with least inter-agent distance below 1 m, especially for the non-Gaussian formulation with 3 components.

for the agents as they reach their goal while avoiding collisions. To compute this mean, we use only the trials that resulted in collision-free trajectories. The mean is computed over 100 trials. We observe that the Gaussian method is (relatively) more conservative than the non-Gaussian method, resulting in a longer path length for most scenarios.

2) *Inter-Agent Distance:* In the ORCA computation, the agent radius is augmented to be 0.5 m, in contrast to the original agent radius of 0.25 m, to provide a safe distance around the agent. Thus, the safe inter-agent distance is 1 m. We compare the Gaussian and non-Gaussian formulations for the number of trials in which the safety threshold distance was compromised. We observe that the non-Gaussian method performs better in this case, and the results are illustrated through a histogram in Fig. 3. We observe that, with an increase in the number of agents in the environment, the inter-agent distance dips below 1 m in multiple trials. However, the non-Gaussian method with 3 Gaussian components performs better, resulting in lower number of such trials.

E. Computation Time

Fig. 4 illustrates the computation time (in milliseconds) of our algorithm for one agent with 1 to 20 neighboring agents in the environment. From our previous work [15] and from our experiments, we observe that considering the closest 10 obstacles provides good collision avoidance performance in most cases. We observe that, on average, our Gaussian method requires ~ 5 ms to compute a control input using our optimization algorithm, while our non-Gaussian method (with 2 Gaussian components) requires ~ 9 ms in scenarios with 4 agents.

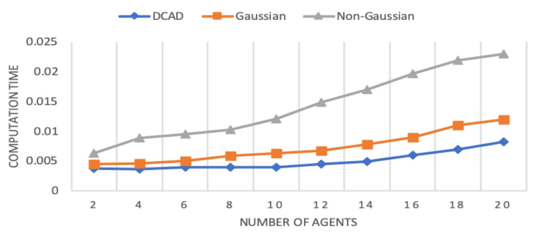


Fig. 4. Control input computation time (ms) for one agent in the presence of 2 to 20 neighboring agents.

VI. Conclusion, Limitation, and Future Work

We present a new probabilistic method for decentralized collision avoidance among quadrotors in a swarm. Our method uses a flatness-based linear MPC to handle quadrotor dynamics and accounts for the state uncertainties using a chance constraint formulation. We describe two approaches to model the chance constraints; the first assumes a Gaussian distribution for the state, while the second approach is more general and can handle non-Gaussian noise using a GMM. Both the methods result in fewer collisions as compared to the deterministic algorithms, but the Gaussian method tends to be more conservative, leading to longer path lengths for the agents. Further, we observed that the non-Gaussian method with 3 Gaussian components demonstrates better performance in terms of satisfying the ORCA constraints, resulting in fewer trials with an inter-agent safety distance lower than 1 m compared to the Gaussian formulation.

Our method has a few limitations. We estimate the distribution of $\mathbf{m}$ using position and velocity samples from a black-box simulator, hence the distribution may not be accurate. The non-Gaussian method is computationally expensive, which affects the rapid re-planning of trajectories. In addition, we do not consider the ego-motion noise, i.e. the noise in implementing the control input. Moreover, our optimization's does not consider the uncertainties in the state. As a part of our future work, we plan to work on faster methods to evaluate the chance constraint for the non-Gaussian, non-parametric case. Additionally, we plan to evaluate our algorithm on physical quadrotors.

References

[1] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A. sequential convex programming approach," in *Proc. IEEE RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 1917–1922.

[2] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, "Towards a swarm of agile micro quadrotors," *Auton. Robots*, vol. 35, no. 4, pp. 287–300, 2013.

[3] J. A. Preiss, W. Hönig, N. Ayanian, and G. S. Sukhatme, "Downwash-aware trajectory planning for large quadrotor teams," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 250–257.

[4] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[5] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2008, pp. 1928–1935.

[6] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robot. Res.*. Springer, 2011, pp. 3–19.

[7] D. Morgan, S.-J. Chung, and F. Y. Hadaegh, "Decentralized model predictive control of swarms of spacecraft using sequential convex programming," *Adv. Astronautical Sci.*, no. 148, pp. 1–20, 2013.

[8] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 1047–1054, Apr. 2017.

[9] K. Khoshelham and S. O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, no. 2, pp. 1437–1454, 2012.

[10] J. S. Park and D. Manocha, "Efficient probabilistic collision detection for non-gaussian noise distributions," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1024–1031, Apr. 2020.

[11] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 696–706, Aug. 2011.

[12] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, "Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 236–243.

[13] H. Zhu and J. Alonso-Mora, "Chance-constrained collision avoidance for mavs in dynamic environments," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 776–783, Apr. 2019.

[14] B. Gopalakrishnan, A. K. Singh, M. Kaushik, K. M. Krishna, and D. Manocha, "Prvo: Probabilistic reciprocal velocity obstacle for multi robot navigation under uncertainty," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 1089–1096.

[15] S. H. Arul and D. Manocha, "DCAD: Decentralized collision avoidance with dynamics constraints for agile quadrotor swarms," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 1191–1198, Apr. 2020.

[16] J. Van Den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 3475–3482.

[17] D. Wolinski, S. J. Guy, A.-H. Olivier, M. Lin, D. Manocha, and J. Pettré, "Parameter estimation and comparative evaluation of crowd simulations," *Comput. Graph. Forum*. Wiley Online Library, vol. 33, no. 2, 2014, pp. 303–312.

[18] M. Rufli, J. Alonso-Mora, and R. Siegwart, "Reciprocal collision avoidance with motion continuity constraints," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 899–912, Aug. 2013.

[19] S. Kim *et al.*, "Brvo: Predicting pedestrian trajectories using velocity-space reasoning," *Int. J. Robot. Res.*, vol. 34, no. 2, pp. 201–217, 2015.

[20] G. Angeris, K. Shah, and M. Schwager, "Fast reciprocal collision avoidance under measurement uncertainty," 2019, *arXiv:1905.12875*.

[21] D. Manocha and J. Demmel, "Algorithms for intersecting parametric and algebraic curves i: Simple intersections," *ACM Trans. Graph.*, vol. 13, no. 1, pp. 73–100, 1994.

[22] H. Zhu and J. Alonso-Mora, "B-uavc: Buffered uncertainty-aware voronoi cells for probabilistic multi-robot collision avoidance," in *Proc. Int. Symp. Multi-Robot Multi-Agent Syst.*, Aug. 2019, pp. 162–168.

[23] P. S. N. Jyotish, B. Gopalakrishnan, A. V. S. S. Bhargav Kumar, A. K. Singh, K. M. Krishna, and D. Manocha, "Reactive navigation under non-parametric uncertainty through hilbert space embedding of probabilistic velocity obstacles," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2690–2697, Apr. 2020.

[24] S. H. Arul and D. Manocha, "Swarmcco: Probabilistic reactive collision avoidance for quadrotor swarms under uncertainty," 2020, *arXiv:2009.07894*.

[25] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 2520–2525.

[26] J. Ferrin, R. Leishman, R. Beard, and T. McLain, "Differential flatness based control of a rotorcraft for aggressive maneuvers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 2688–2693.

[27] A. Charnes and W. W. Cooper, "Chance-constrained programming," *Manage. Sci.*, vol. 6, no. 1, pp. 73–79, 1959.

[28] A. Prékopa, *Stochastic Programming*. Springer Science & Business Media, 2013, vol. 324.

[29] Z. Hu, W. Sun, and S. Zhu, "Chance Constrained Programs With Mixture Distributions," 2018.