Trajectory Generation by Chance-Constrained Nonlinear MPC With Probabilistic Prediction

Xiaoxue Zhang, Jun Ma, Zilong Cheng, Sunan Huang, Shuzhi Sam Ge, *Fellow, IEEE*, and Tong Heng Lee

***Abstract*—Continued great efforts have been dedicated toward high-quality trajectory generation based on optimization methods; however, most of them do not suitably and effectively consider the situation with moving obstacles; and more particularly, the future position of these moving obstacles in the presence of uncertainty within some possible prescribed prediction horizon. To cater to this rather major shortcoming, this work shows how a variational Bayesian Gaussian mixture model (vBGMM) framework can be employed to predict the future trajectory of moving obstacles; and then with this methodology, a trajectory generation framework is proposed which will efficiently and effectively address trajectory generation in the presence of moving obstacles, and incorporate the presence of uncertainty within a prediction horizon. In this work, the full predictive conditional probability density function (PDF) with mean and covariance is obtained and, thus, a future trajectory with uncertainty is formulated as a collision region represented by a confidence ellipsoid. To avoid the collision region, chance constraints are imposed to restrict the collision probability, and subsequently, a nonlinear model predictive control problem is constructed with these chance constraints. It is shown that the proposed approach is able to predict the future position of the moving obstacles effectively; and, thus, based on the environmental information of the probabilistic prediction, it is also shown that the timing of collision avoidance can be earlier than the method without prediction. The tracking error and distance to obstacles of the trajectory with prediction are smaller compared with the method without prediction.**

***Index Terms*—Chance constraint, Gaussian mixture model, model predictive control (MPC), trajectory prediction, variational inference.**

I. Introduction

TRAJECTORY generation is certainly one of the critical component technologies for autonomous robots [1]–[3]; and it involves not only a path planning problem to find a sequence of valid configurations that moves a mobile robot but also refers to the larger problem regarding how to move along the path in various real-world practical situations. Pertinent to addressing these mathematical formulations involving such substantial and possibly difficult equality and inequality constraints [4]–[7], it is noteworthy that model predictive control (MPC) is an effective technique in addressing various constraints as part of the control synthesis problem [8]–[10]. However, various drawbacks exist, such as requiring the more restrictive assumption that the unmanned aerial vehicle (UAV) moves on a 2-D surface instead of a 3-D environment [11], and lack of consideration of certain environmental factors (obstacles and their motion) [8]. Overall, the prediction of surrounding moving obstacles is a rather challenging problem due to a large number of factors that influence the future states of robots.

In the existing literature, various approaches are applied in a typical trajectory prediction task, such as Bayesian network [12]; hidden Markov models (HMMs) [13]; Monte Carlo simulation [14]; Kalman filters [15]; long-short temporal memory (LSTM) [16], [17]; generative adversarial networks (GANs) [18]–[20]; etc. While all these efforts indicate great possibilities and promise, yet at present stages of development, various drawbacks exist, such as certain methodologies requiring rather prohibitively high computational resources (memory-bandwidth computation) to train these networks suitably fast, and difficulties with the gap between the parameter space and the function space. Some recent research works, on the other hand, reveal the rather significant improvements and advantages with the incorporation and use of a Bayesian network approach [12]. Here, the probabilistic method gives a probability distribution over the training trajectories, and it additionally provides the conditional distribution of the future horizon given partial history trajectory snippets [21]. This method also considers a degree of uncertainty for future predictions.

With all of the above descriptions as a backdrop, in this work, we develop a chance-constrained nonlinear MPC approach to generate the suitable required collision-free trajectory. We formulate the predicted distribution based on a variational Bayesian Gaussian mixture model (vBGMM) framework as probabilistic chance constraints for the MPC problem; and further solve the resulting nonlinear MPC problem characterizing the collision-free trajectory generation task.

Manuscript received January 13, 2020; revised May 14, 2020 and August 4, 2020; accepted October 12, 2020. Date of publication November 24, 2020; date of current version June 23, 2021. This article was recommended by Associate Editor Z. Li. *(Corresponding author: Jun Ma.)*

Xiaoxue Zhang, Zilong Cheng, Shuzhi Sam Ge, and Tong Heng Lee are with the NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, Singapore 119077, and also with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: xiaoxuezhang@u.nus.edu; zilongcheng@u.nus.edu; samge@nus.edu.sg; eleleeth@nus.edu.sg).

Jun Ma is with the Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: jun.ma@berkeley.edu).

Sunan Huang is with the Temasek Laboratories, National University of Singapore, Singapore 117411 (e-mail: tslhs@nus.edu.sg).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2020.3032711.

Digital Object Identifier 10.1109/TCYB.2020.3032711

The key significant contribution of this article is essentially two-fold: both the predicted uncertainty and potential collision are considered during the prediction horizon in the nonlinear MPC problem. Therefore, with this new and significant development here, our resulting solution simultaneously ensures that first, the risk of a collision caused by parametric uncertainty and sensor noise is greatly decreased; and second, the required suitable collision-free trajectory can also be generated in advance. It is noteworthy that compared to the existing MPC-based methods without prediction of moving obstacles, our proposed approach can significantly and effectively improve the quality of the generated trajectories.

The remainder of this article is organized as follows. Section II first lays out the details of the key basis of our proposed process of trajectory prediction by using the vBGMM framework. Section III then presents the formulation and development of our proposed trajectory generation approach (with prediction) to efficiently and effectively address trajectory generation in the presence of moving obstacles. Here, uncertainty is incorporated as chance constraints, and an appropriate nonlinear MPC problem is formulated with these constraints. Then, in Section IV, a case study on the trajectory generation problem for a quadcopter is given. Finally, the conclusion of this work is given in Section V.

II. Probabilistic Prediction

The purpose of this section is to show the prediction of the future trajectories for moving obstacles. Since this probabilistic modeling method requires a probability density function (PDF), we can infer the approximated PDF based on the training data. In this section, a joint distribution of history and future data in the training trajectories will be inferred based on vBGMM. Then, the conditional PDF of the future trajectory of the test data can be obtained by computing the statistical parameters.

A. Trajectory Representation

Chebyshev decomposition of trajectories is applied to represent the characteristics of trajectories. The Chebyshev polynomial of a degree of n is defined as

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 \\ &\vdots \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x). \end{aligned} \tag{1}$$

The Chebyshev polynomial T_n is orthogonal in the interval $[-1, 1]$ and has n zeros in this interval, which means the error between the function we need to approximate and the Chebyshev approximation is close to the optimal nth-degree polynomial. To approximate any arbitrary function $f(x)$, the Chebyshev coefficients a_n can be calculated by using

$$a_n = \frac{2}{N}\sum_{k=0}^{N-1} f(x_k)T_n(x_k) \tag{2}$$

where x_k are N zeros of $T_N(x)$. $\boldsymbol{a} = \begin{bmatrix} a_0 & a_1 & \cdots & a_N \end{bmatrix}$ will be used as input feature to train and predict the probabilistic distribution. Denote x, y, and z as the standard Cartesian coordinates and v, θ, and ϕ as the spherical coordinates. For appropriately better capture of the notation for the rotation in a trajectory, we use v, θ, and ϕ to characterize the trajectory.

B. Variational Bayesian Inference

The probabilistic trajectory prediction can be formulated as an estimation of the conditional distribution of predicted positions given the history positions of the moving obstacles. This conditional distribution is given by

$$\Pr(\boldsymbol{a}_f \mid \boldsymbol{a}_h) = \Pr(a_{v,f}, a_{\theta,f}, a_{\phi,f} \mid a_{v,h}, a_{\theta,h}, a_{\phi,h}) \tag{3}$$

where $\boldsymbol{a}_h$ and $\boldsymbol{a}_f$ are the Chebyshev approximation coefficients vectors corresponding to history trajectories and future trajectories. All of the subscripts $\cdot_f$ and $\cdot_h$ denote the parameters regarding the future and history, respectively.

First, the joint distribution $\Pr(\boldsymbol{a}_f, \boldsymbol{a}_h)$ can be modeled by GMM which comprises a number of component Gaussian functions to provide a multimodel density function. Some previous researches apply some maximum-likelihood solutions or 3-σ confidence ellipses to predict the future trajectory [22], [23]. The Bayesian methodology, that is, variational inference, can be used to estimate this GMM and provide a lower bound on the approximation error [24]. Variational Bayesian inference has outstanding generation performance and can conquer some shortcomings of these previous methods, such as singularity in the covariance matrix, overfitting, and sensibility to the outliers. In this method, the whole conditional predicted distribution can be obtained given the prior distributions of the parameters. In the Bayesian setting, we consider a prior on the model parameters and aim to infer their posterior distribution by

$$\begin{aligned} \Pr(\boldsymbol{\pi}) &= \mathrm{Dir}(\boldsymbol{\pi}|\alpha_0) \\ &= C(\alpha_0)\prod_{k=1}^{K}\pi_k^{\alpha_0-1} \end{aligned} \tag{4a}$$

$$\begin{aligned} \Pr(\boldsymbol{\mu}, \boldsymbol{\Lambda}) &= p(\boldsymbol{\mu}|\boldsymbol{\Lambda})p(\boldsymbol{\Lambda}) \\ &= \prod_{k=1}^{K}\mathcal{N}\left(\boldsymbol{\mu}_k|\mathbf{m}_0, (\beta_0\boldsymbol{\Lambda}_k)^{-1}\right)\mathcal{W}(\boldsymbol{\Lambda}_k|\mathbf{W}_0, \nu_0) \end{aligned} \tag{4b}$$

where K is the number of mixture components, and Dir means the Dirichlet distribution, which is used as the conjugate prior of the multinomial distribution of weights $\boldsymbol{\pi}$, α_0 and $C(\alpha_0)$ are the set of concentration parameters and the normalization constant of the Dirichlet distribution, respectively. The parameter α_0 can be considered as the prior number of observations connected to the components of the mixture model. If the value of α_0 is larger, the posterior distribution is more influenced by the prior instead of the data. $\mathcal{N}$ and $\mathcal{W}$ denote the Normal and Wishart distribution. An independent Normal–Wishart distribution is used as the conjugate prior distribution when both means and precision of Gaussian mixture components $\boldsymbol{\mu}$ and $\boldsymbol{\Lambda}$ are unknown, as shown in (4b). $\mathbf{W}_0$ and $\mathbf{m}_0$ are the initial priors for precisions and means, and β_0 and ν_0 are the

initial scaling factor and degree of freedom of the Wishart distribution, respectively.

It seems infeasible to evaluate the posterior distribution because the dimensionality of the latent space is too high, and the posterior distribution is too complex to have an analytically tractable solution. Therefore, variational Bayesian inference is useful to obtain the approximated parameters of the posterior distribution. Similar to [21], we also use the variational Bayesian expectation–maximization algorithm [25] to infer the posterior distribution and obtain the approximated parameters of this distribution. The predictive density distribution for a new variable $\boldsymbol{a}$ of the given observed data is a mixture of student's t-distribution [25], which can be calculated by

$$\begin{aligned}\Pr(\boldsymbol{a}_f, \boldsymbol{a}_h) &= \frac{\sum_{k=1}^{K} \alpha_k T(\boldsymbol{a}_f, \boldsymbol{a}_h | \mathbf{m}_k, \boldsymbol{L}_k, \nu_k + 1 - D)}{\sum_{k=1}^{K} \alpha_k} \\ \boldsymbol{L}_k &= \frac{(\nu_k + 1 - D)\beta_k}{1 + \beta_k} \mathbf{W}_k \end{aligned} \tag{5}$$

where D is the dimension of data, T denotes the student's t-distribution with mean $\boldsymbol{m}_k$ and precision $\boldsymbol{L}_k$ of the kth component, and α_k, β_k, and ν_k are the mixing parameter, scaling factor, and degree of freedom of the kth component, respectively. The variational lower bound can be used to determine the posterior distribution over K components in the mixture model. A suitable value of K can be determined by treating the mixing coefficients $\boldsymbol{\pi}$ as parameters and making point estimation by maximizing the lower bound with respect to $\boldsymbol{\pi}$, rather than computing the distribution by fully Bayesian rule. Hence, re-estimation of the $\boldsymbol{\pi}$ executes after updating the factorized distribution over other parameters except for π_k will lead to sparsity given any initial value of K.

At this point, as part of our development to show the prediction of the future trajectories for moving obstacles, it is useful to state the following intermediate result on the density probability of the predicted future trajectory (of the observed history trajectory).

Lemma 1: Based on this joint distribution $\Pr(\boldsymbol{a}_f, \boldsymbol{a}_h)$, the density probability of predicted future trajectory of the observed history trajectory can be calculated by computing the conditional distribution $\Pr(\boldsymbol{a}_f \mid \boldsymbol{a}_h)$ as

$$\begin{aligned}\Pr(\boldsymbol{a}_f|\boldsymbol{a}_h) &= \frac{\sum_{k=1}^{K} \tilde{\alpha}_k T(\boldsymbol{a}_f|\boldsymbol{a}_h, \mathbf{m}_{k,f|h}, \mathbf{L}_{k,f|h}, \nu_k + 1)}{\sum_{k=1}^{K} \tilde{\alpha}_k} \\ \tilde{\alpha}_k &= \frac{\alpha_k T(\boldsymbol{a}_h|\mathbf{m}_{k,h}, \mathbf{L}_{k,h}, \nu_k + 1 - D)}{\sum_{j=1}^{K} \alpha_j T(\boldsymbol{a}_h|\mathbf{m}_{j,h}, \mathbf{L}_{j,h}, \nu_k + 1 - D)} \\ \mathbf{m}_{k,f|h} &= \mathbf{m}_{k,f} + \boldsymbol{\Sigma}_{k,fh}\boldsymbol{\Sigma}_{k,hh}^{-1}(\boldsymbol{a}_h - \mathbf{m}_{k,h}) \\ \boldsymbol{L}_{k,f|h}^{-1} &= \left(1 + (\boldsymbol{a}_h - \mathbf{m}_{k,h})^T \frac{\Sigma_{k,hh}^{-1}}{\nu_{k,f|h}}(\boldsymbol{a}_h - \mathbf{m}_{k,h})\right) \\ &\quad \left(\boldsymbol{\Sigma}_{k,ff} - \boldsymbol{\Sigma}_{k,fh}\boldsymbol{\Sigma}_{k,hh}^{-1}\boldsymbol{\Sigma}_{k,hf}\right)\frac{\nu_k + 1 - D}{\nu_k - 1} \\ \boldsymbol{\Sigma}_{k,f|h} &= \frac{\nu_k - 1}{\nu_k + 1}\boldsymbol{L}_{k,f|h}^{-1} \end{aligned} \tag{6}$$

where the notation $\cdot_{f|h}$ means the corresponding parameters in the conditional distribution of future data given history data, and $\boldsymbol{m}_k = \begin{bmatrix}\boldsymbol{m}_{kh} \\ \boldsymbol{m}_{kf}\end{bmatrix}$ and $\boldsymbol{\Sigma}_k = \begin{bmatrix}\boldsymbol{\Sigma}_{k,hh} & \boldsymbol{\Sigma}_{k,hf} \\ \boldsymbol{\Sigma}_{k,fh} & \boldsymbol{\Sigma}_{k,ff}\end{bmatrix}$ are the partition of means and covariances of this mixture student's t-distribution. The subscripts $\cdot_{hh}$, $\cdot_{hf}$, $\cdot_{fh}$, and $\cdot_{ff}$ represent the parameter with respect to $\cdot_{\boldsymbol{a}_h,\boldsymbol{a}_h}$, $\cdot_{\boldsymbol{a}_h,\boldsymbol{a}_f}$, $\cdot_{\boldsymbol{a}_f,\boldsymbol{a}_h}$, and $\cdot_{\boldsymbol{a}_f,\boldsymbol{a}_f}$, respectively.

Proof: Define $\boldsymbol{X} = \begin{bmatrix}\boldsymbol{a}_h \\ \boldsymbol{a}_f\end{bmatrix}$, $\boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{m}_h \\ \boldsymbol{m}_f\end{bmatrix}$, and $\boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_{hh} & \boldsymbol{\Sigma}_{hf} \\ \boldsymbol{\Sigma}_{fh} & \boldsymbol{\Sigma}_{ff}\end{bmatrix}$. According to the mixture representation, the characteristic function of $\boldsymbol{X}$ following a multivariate students' t-distribution is given by:

$$\begin{aligned}\phi_{\boldsymbol{X}}(\boldsymbol{t}) &= \mathbb{E}\left(e^{i\boldsymbol{t}^T\boldsymbol{X}}\right) \\ &= e^{i\boldsymbol{t}^T\boldsymbol{\mu}} \frac{\left\|(\nu\boldsymbol{\Sigma})^{1/2}\boldsymbol{t}\right\|^{\nu/2}}{2^{\nu/2-1}\Gamma(\nu/2)} K_{\nu/2}\left(\left\|(\nu\boldsymbol{\Sigma})^{1/2}\boldsymbol{t}\right\|\right) \end{aligned} \tag{7}$$

where $K_{\nu/2}(\|(\nu\boldsymbol{\Sigma})^{1/2}\boldsymbol{t}\|)$ is the Macdonald function with order $\nu/2$ and argument $\|(\nu\boldsymbol{\Sigma})^{1/2}\boldsymbol{t}\|$. Using (7), we can obtain $\boldsymbol{a}_h \sim T(\boldsymbol{m}_h, [\nu/(\nu-2)]\boldsymbol{\Sigma}_{hh})$. Then, the conditional distribution of $\boldsymbol{a}_f$ given $\boldsymbol{a}_h$ is $T(\boldsymbol{m}_{f|h}, \boldsymbol{\Sigma}_{f|h}, \nu_{f|h})$. Therefore, the conditional distribution of mixture students' t-distribution can be written as (6). ■

Remark 1: Overfitting is not a concern when using variational inference as it can find the optimal cluster components K given an initial value.

The posterior joint distribution and its parameters can be obtained after the training process, and then the conditional distribution will be used to predict the future trajectory in the prediction process based on the parameters calculated in the training process. The derived conditional distribution (6) defines a conditional PDF of the future trajectories whose mean and covariance can be evaluated by

$$\begin{aligned}\boldsymbol{\mu} &= \begin{bmatrix}\boldsymbol{\mu}_v & \boldsymbol{\mu}_\theta & \boldsymbol{\mu}_\phi\end{bmatrix} = \sum_{k=1}^{K} \tilde{\alpha}_k \boldsymbol{m}_k \\ \boldsymbol{\Sigma} &= \begin{bmatrix}\boldsymbol{\Sigma}_{v,v} & \boldsymbol{\Sigma}_{v,\theta} & \boldsymbol{\Sigma}_{v,\phi} \\ \boldsymbol{\Sigma}_{\theta,v} & \boldsymbol{\Sigma}_{\theta,\theta} & \boldsymbol{\Sigma}_{\theta,\phi} \\ \boldsymbol{\Sigma}_{\phi,v} & \boldsymbol{\Sigma}_{\phi,\theta} & \boldsymbol{\Sigma}_{\phi,\phi}\end{bmatrix} \\ &= \sum_{k=1}^{K} \tilde{\alpha}_k\left(\boldsymbol{\Sigma}_k + (\boldsymbol{m}_k - \boldsymbol{m})(\boldsymbol{m}_k - \boldsymbol{m})^T\right). \end{aligned} \tag{8}$$

After training, the predicted Chebyshev coefficients are distributed with $\boldsymbol{a}_{(\cdot)} \sim \mathcal{N}(\boldsymbol{\mu}_{(\cdot)}, \boldsymbol{\Sigma}_{(\cdot),(\cdot)})$, where $\boldsymbol{\mu}_{(\cdot)}$ and $\boldsymbol{\Sigma}_{(\cdot),(\cdot)}$ are the corresponding mean and covariance for each variable v, θ, and ϕ. Thus, the mean and covariance of v, θ, and ϕ can be evaluated by reconstructing this Chebyshev approximation based on the coefficients $\boldsymbol{a}_{(\cdot)}$. Then, it is followed by a transformation function from spherical coordinate v, θ, and ϕ to Cartesian coordinate x, y, and z.

III. Nonlinear MPC With Chance Constraints

In this section, a nonlinear MPC problem is formulated and then solved appropriately. First, for the purpose of collision avoidance, the obstacle region is represented by ellipsoids and further transformed into chance constraints. Then, we reformulate the collision-avoidance chance constraints as deterministic

constraints, which are integrated into the MPC problem. Then, the stability analysis is provided, and an optimization algorithm is presented to solve this problem.

A. Obstacle Region

After probabilistic trajectory prediction, we can obtain the means and covariances of the future trajectory, which can be formulated as a predicted region where the host agent needs to avoid, called an obstacle region $\mathcal{I}$. For the ith moving obstacle, assume its future position probabilistically lies in the obstacle region $\mathcal{I}_i$ at time t which is based on the mean $\boldsymbol{\mu}_i(t) = \begin{bmatrix}\mu_x & \mu_y & \mu_z\end{bmatrix}^T$ and covariance $\boldsymbol{\Sigma}_i(t) \in \mathbb{R}^{3\times 3}$ with respect to time t. For simplicity, the variable t will be neglected for the following description in this section. In such a way, we can assume that the future position of the ith moving obstacle can be represented as $\boldsymbol{p}_{i,f} \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ at time t.

Remark 2: Since the covariance matrix $\boldsymbol{\Sigma}_i$ is real symmetric and positive semidefinite, the eigenvalues are real, and there exists an orthogonal matrix $\mathbf{Q}_i$ formed by eigenvectors of $\boldsymbol{\Sigma}_i$, we can carry out the spectral decomposition for the covariance matrix $\boldsymbol{\Sigma}_i$ as

$$\boldsymbol{\Sigma}_i = \mathbf{Q}_i \boldsymbol{\Lambda}_i \mathbf{Q}_i^T \tag{9}$$

where $\boldsymbol{\Lambda}_i = \text{diag}(\lambda_j)$, $j = 1, 2, 3$, where λ_j is sorted in descending order with $\lambda_1 \geq \lambda_2 \geq \lambda_3$. Here, j means each dimension in the environment.

At this juncture, it is pertinent to state two key intermediate results (on ellipsoid construction, and on approximate scaling factor computation) that are significant essential parts in the development which follows the nonlinear MPC with chance constraints methodology. Thus first, note the following first intermediate result on ellipsoid construction.

Lemma 2: Ellipsoid can be constructed from the transformation of a unit sphere by first stretching with a ratio of $\sqrt{\lambda_i}$ along each axis, then rotating the ellipsoid by $\mathbf{Q}_i$ and a final translation of the distribution center $\mathbf{m}_i$ according to the following inverse Mahalanobis transformation:

$$\mathcal{I}_i = \mathbf{Q}_i \boldsymbol{\Lambda}_i^{\frac{1}{2}} \mathbf{Q}_i^T \mathbf{u} + \boldsymbol{\mu}_i \tag{10}$$

where $\mathbf{u} \sim \mathcal{N}(0, \boldsymbol{I}_3)$ is a unit sphere with normal distribution in three dimensions. In this work, $\boldsymbol{I}_n$ denotes the identity matrix with the size of $n \times n$.

Proof: Mapping a unit sphere by the square root of the covariance matrix, $\boldsymbol{\Sigma}_i^{(1/2)}$ determines an ellipsoid whose principle semiaxes rely on the eigenvalues of this matrix and the orientation is related to the corresponding eigenvectors. In order to represent this ellipsoid graphically, the Mahalanobis transformation can be used to eliminate the correlation between the variables and to standardize each variable with variance [26]. Therefore, an ellipsoid can be constructed from the transformation of a unit sphere, according to the inverse Mahalanobis transformation (10). ■

Next here, note the following intermediate result on approximate scaling factor computation.

Lemma 3: The approximate scaling factor r can be computed by

$$F(r) = \mathscr{P}(r) - \tilde{\varphi}$$
$$\dot{F}(r) = \dot{\mathscr{P}}(r) = \sqrt{\frac{2}{\pi}} \exp\left(-\frac{r^2}{2}\right) + \frac{\exp\left(-\frac{r^2}{2}\right)}{\sqrt{2}\Gamma\left(\frac{3}{2}\right)}(r^2 - 1). \tag{11}$$

Proof: Based on Lemma 2, the square of the Mahalanobis distance (scaling factor r) of the probable position $\boldsymbol{p}_{i,f}$ to its mean $\boldsymbol{\mu}_i$ can be calculated by

$$r^2 = \left(\boldsymbol{p}_{i,f} - \mathbf{m}_i\right)^T \boldsymbol{\Sigma}_i^{-1} \left(\boldsymbol{p}_{i,f} - \mathbf{m}_i\right). \tag{12}$$

Substituting (9) and (10) into (12), we can obtain that the magnified ellipsoid with ratio r relies on the chi-square χ^2 distribution with a degree of freedom $\varrho = 3$, as shown in

$$\Pr\left(r^2 \leq \chi^2_{\varrho=3,p}\right) = \tilde{\varphi} \tag{13}$$

which can be represented by

$$\Pr\left(\left(\boldsymbol{a}_{i,f} - \boldsymbol{\mu}_i\right)^T \boldsymbol{\Sigma}_i^{-1} \left(\boldsymbol{a}_{i,f} - \boldsymbol{\mu}_i\right) \leq r^2\right) = \Pr\left(\mathbf{u}^T \mathbf{u} \leq r^2\right).$$

The confidence probability for an arbitrary ellipsoid with any factor r is

$$\begin{aligned}\mathscr{P}(r) &= \Pr\left(\mathbf{u}^T \mathbf{u} \leq r^2\right) \\ &= \iiint (2\pi)^{-\frac{3}{2}} \exp\left(\frac{u_1^2 + u_2^2 + u_3^2}{2}\right) du_1 du_2 du_3 \\ &= \text{erf}\left(\frac{r}{\sqrt{2}}\right) - \left(\frac{r}{\sqrt{2}}\right) \frac{\exp\left(-\frac{r^2}{2}\right)}{\Gamma\left(\frac{3}{2}\right)}\end{aligned} \tag{14}$$

where

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2) dt$$

is the standard error function, and Γ is the gamma function. Given the confidence level, the scaling factor of the ellipsoid can be calculated by the cumulative distribution function $F(r)$ and its derivatives $\dot{F}(x)$ as (11). ■

Then, r can be solved by iterative Newton-based methods based on Lemma 3. In fact, the confidence ellipsoids in different confidence levels can be obtained, which form the obstacle region $\mathcal{I}_i$ for the ith moving obstacle. In such a case, the scaling factor $r = 2.5003$, 2.7955, and 3.3682 when the confidence level $p = 90\%$, 95%, and 99%, respectively.

B. Chance Constraint

Assume there are n_o moving obstacles the host robot can detect at the moment t. Checking whether there is a collision happening between the host robot and a moving obstacle i requires to compute the minimum distance between the current position of robot $\boldsymbol{p}(t)$ and the collision region of the ith moving obstacle $\mathcal{I}_i$. Notably, $\boldsymbol{p}(t)$ is the part of the state variable $\boldsymbol{x}(t)$. The collision condition of the host robot with respect to the moving obstacle i at time t is defined as

$$\mathbb{C}_i^t \coloneqq \left\{\boldsymbol{p}(t) \mid \left\|\boldsymbol{p}(t) - \hat{\boldsymbol{p}}_i(t)\right\| \leq d_{\text{safe}},\ \hat{\boldsymbol{p}}_i(t) \in \mathcal{I}_i(t)\right\} \tag{15}$$

where $\hat{\boldsymbol{p}}_i(t)$ denotes the possible position of the ith obstacle in time t, d_{safe} means the predefined safety distance between the host agent and the moving obstacles, and $\|\cdot\|$ is the Euclidean norm. Since the predicted positions are represented by a probability distribution, the predicted collision-avoidance constraints can be formulated in a probabilistic manner, which are so-called chance constraints

$$\Pr\big(\boldsymbol{p}(t) \in \mathbb{C}_i^t\big) \leq \varphi, \quad i \in \mathbb{N}_{n_o} \tag{16}$$

where φ is the probability threshold for the robot-obstacle collision, the set $\mathbb{N}_{n_o} = \{1, 2, \ldots, n_o\}$ and n_o is the number of moving obstacles the robot can detect. At this point, it is pertinent to also state the following intermediate result.

Lemma 4: Given any matrix $\mathbf{A}$ and scalar b, for a multivariate random variable $\mathbf{X}(t)$ corresponding to the mean $\boldsymbol{\mu}(t)$ and covariance $\boldsymbol{\Sigma}(t)$, the chance constraint

$$\Pr\big(\mathbf{A}^T\mathbf{X}(t) < b\big) \leq \varphi \tag{17}$$

is equivalent to a deterministic constraint

$$\mathbf{A}^T\boldsymbol{\mu}(t) - b \geq \eta \tag{18}$$

where $\eta = \sqrt{2\mathbf{A}^T\boldsymbol{\Sigma}(t)\mathbf{A}}\,\text{erf}^{-1}(1-2\varphi)$ and φ is the predefined allowable probability threshold of collision.

Proof: Given a univariate Gaussian random variable $X \sim \mathcal{N}(\mu, \sigma^2)$ with known variance, according to the definition of PDF, we have that $\Pr(X < 0) \leq \varphi$ is equal to $\mu \geq \eta$, where $\eta = \sqrt{2}\sigma\,\text{erf}^{-1}(1-2\varphi)$.

In terms of multivariate Gaussian random variable $X(t) \sim \mathcal{N}(\mu(t), \Sigma(t))$ at time t, set a univariate random variable $Y(t)$ is the perpendicular distance between the plane $\mathbf{A}^T\mathbf{X}(t) = b$ and the point $X(t)$, and then the event $\mathbf{A}^T\mathbf{X}(t) < b$ is equal to $Y(t) < 0$. Based on the relationship between $Y(t)$ and $X(t)$, we have $Y(t) \sim \mathcal{N}(\mu_Y, \sigma_Y)$, where $\mu_Y = \mathbf{A}^T\mu(t) - b$ and $\sigma_Y = \sqrt{\mathbf{A}^T\Sigma(t)\mathbf{A}}$. Here, $\Pr(\mathbf{A}^T\mathbf{X}(t) < b) \leq \varphi$ is equal to $\Pr(Y(t) < 0) \leq \varphi$. Applying the above result of the univariate Gaussian random variable $Y(t)$, we can obtain $\mu_Y \geq \eta$, where $\eta = \sqrt{2}\sigma_Y\,\text{erf}^{-1}(1-2\varphi)$. Therefore, (17) is equivalent to (18). ■

Particularly, key in our work here is the appropriately interesting utilization of the notion of chance constraints, where (as also mentioned earlier) the predicted collision-avoidance constraints can be formulated as chance constraints. Along this line then, the following main result is of particular importance.

Theorem 1: The chance constraint (16) can be reformulated as a deterministic constraint as

$$\begin{aligned}&\boldsymbol{\kappa}_i^T(t)\big(\boldsymbol{p}(t) - \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))\big)\\ &\quad\geq \sqrt{2\boldsymbol{\kappa}_i^T(t)\boldsymbol{\Sigma}(t)\boldsymbol{\kappa}_i(t)}\,\text{erf}^{-1}(1-2\varphi) + d_{\text{safe}}^2\end{aligned} \tag{19}$$

where

$$\boldsymbol{\kappa}_i(t) = \frac{\boldsymbol{p}(t) - \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))}{\big\|\boldsymbol{p}(t) - \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))\big\|}$$

is the slope of the line connecting $\boldsymbol{p}(t)$ and $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$ and is perpendicular to the tangent plane of $\mathcal{I}_i(t)$ at $\boldsymbol{p}(t)$, and the point $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$ is the projection of $\boldsymbol{p}(t)$ onto the set $\mathcal{I}_i(t)$.

Proof: As mentioned in Section III-A, the obstacle region $\mathcal{I}_i(t)$ for any moving obstacle i at each time moment t can be described as an ellipsoid according to its mean $\boldsymbol{\mu}_i(t)$ and covariance $\boldsymbol{\Sigma}_i(t)$. Obviously, this ellipsoid region $\mathcal{I}_i(t)$ is a convex set. We can find a point $\hat{\boldsymbol{p}} \in \mathcal{I}_i$ which is the closest point from the given position $\boldsymbol{p}(t)$ of the host robot. The closest point can be represented in the projection form

$$\hat{\boldsymbol{p}}(t) = \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t)) \tag{20}$$

where $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t)) \coloneqq \min\{(1/2)\|\hat{\boldsymbol{p}}(t) - \boldsymbol{p}(t)\|^2 \mid \hat{\boldsymbol{p}}(t) \in \mathcal{I}_i\}$ is the projection of $\boldsymbol{p}(t)$ onto $\mathcal{I}_i(t)$.

Then, the tangent plane over the closest point $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$ is perpendicular to the line from $\boldsymbol{p}(t)$ to $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$, which can be represented by

$$\boldsymbol{\kappa}_i^T(t)\big(\boldsymbol{p}(t) - \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))\big) = d_{\text{safe}}^2. \tag{21}$$

Therefore, the collision region can be enlarged as a half space

$$\hat{\mathbb{C}}_i(t) \coloneqq \Big\{\boldsymbol{p} \mid \boldsymbol{\kappa}_i^T(t)\big(\boldsymbol{p}(t) - \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))\big) \leq d_{\text{safe}}^2\Big\}. \tag{22}$$

It is obvious that the collision region $\mathcal{I}_i \subset \hat{\mathbb{C}}_i(t)$ and, thus, $\Pr(\boldsymbol{p}(t) \in \mathcal{I}_i(t)) \leq \Pr(\boldsymbol{p}(t) \in \hat{\mathbb{C}}_i(t))$. The original chance constraints (16) can be relaxed as (22) and reformulated as deterministic constraints (19), based on Lemma 4. ■

Remark 3: The closest point $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$ can be calculated by solving the following optimization problem:

$$\begin{aligned}&\min_{\boldsymbol{y}} \ \big(\boldsymbol{y} - \boldsymbol{p}_{i,f}\big)^T\big(\boldsymbol{y} - \boldsymbol{p}_{i,f}\big)\\ &\text{s.t.} \ \big(\boldsymbol{y} - \boldsymbol{\mu}_i\big)^T\boldsymbol{\Sigma}_i^{-1}\big(\boldsymbol{y} - \boldsymbol{\mu}_i\big) \leq r^2.\end{aligned} \tag{23}$$

Obviously, if $\boldsymbol{p}_{i,f}$ is inside $\mathcal{I}_i$, then $\boldsymbol{y} = \Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t)) = \boldsymbol{p}_{i,f}$ and the distance between the closest point and the position of the host agent $\text{dist}(\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t)), \boldsymbol{p}_{i,f}) = 0$. Otherwise, $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t)) = \boldsymbol{y}$ is on the boundary of $\mathcal{I}_i$. This problem can be transformed into a quadratic minimization problem and, thus, the Lagrangian method can be used [27], [28]. Define the Lagrangian function

$$\mathcal{L} = \big(\boldsymbol{y} - \boldsymbol{p}_{i,f}\big)^T\big(\boldsymbol{y} - \boldsymbol{p}_{i,f}\big) + \lambda\Big(\big(\boldsymbol{y} - \boldsymbol{\mu}_i\big)^T\boldsymbol{\Sigma}_i^{-1}\big(\boldsymbol{y} - \boldsymbol{\mu}_i\big) - r^2\Big) \tag{24}$$

where λ is the Lagrange multiplier. The Karush–Kuhn–Tucker (KKT) conditions are

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial\boldsymbol{y}} &= 2\big(\boldsymbol{y} - \boldsymbol{p}_{i,f}\big) + 2\lambda\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{y} - \mathbf{m}_i) = 0\\ \frac{\partial\mathcal{L}}{\partial\lambda} &= \big(\boldsymbol{y} - \boldsymbol{\mu}_i\big)^T\boldsymbol{\Sigma}_i^{-1}\big(\boldsymbol{y} - \boldsymbol{\mu}_i\big) - r^2 = 0.\end{aligned} \tag{25}$$

Hence, the optimal solution $\boldsymbol{y}^*$, that is, $\Pi_{\mathcal{I}_i(t)}(\boldsymbol{p}(t))$ can be obtained by solving (25) via gradient-based methods. In some cases, the projection point can be replaced by the mean of the predicted probabilistic position for the obstacle, in order to simplify the computational process.

C. Problem Formulation

Based on the trajectory prediction and chance constraints reformulation, we can interpret the probabilistic prediction as deterministic constraints. Therefore, we can formulate an MPC problem to find the (sub)optimal control input sequence to generate (sub)optimal trajectory for the host robot, while considering the obstacles' future positions.

1) Dynamic Model: Here, we consider a nonlinear dynamic model for an agent, which can be written as

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad k \in \mathbb{N}_{N-1} \tag{26}$$

where $\boldsymbol{x}_k \in \mathbb{R}^n$ and $\boldsymbol{u}_k \in \mathbb{R}^m$ denote the state variables and control inputs of this dynamic model at time step k, f represents the dynamics, and $\mathbb{N}_{N-1}$ is the set of non-negative integers. This nonlinear dynamic model can be approximated by a linear time-variant system model with time-variant matrices $\boldsymbol{A}_t \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B}_t \in \mathbb{R}^{n\times m}$.

2) Constraints: Some physical limitations need to be considered when computing the optimal control inputs, where are shown in

$$\boldsymbol{x}_{k+1} \in \mathcal{X}, \quad \boldsymbol{u}_k \in \mathcal{U} \tag{27}$$

where $\mathcal{X} \in \mathbb{R}^n$ and $\mathcal{U} \in \mathbb{R}^m$ denote the bounded set of $\boldsymbol{x}$ and $\boldsymbol{u}$, respectively.

In order to avoid the potential collision with obstacles, the reformulated chance constraints can be embedded as part of the constraints of the MPC problem, as (19).

For other static obstacles, the generated trajectory should guarantee that the distance between the current location and obstacles is greater than the predefined safe distance d_{safe}, as shown in

$$\left\| L\boldsymbol{x}_k - L\boldsymbol{x}^k_{\text{obs}} \right\| \geq d_{\text{safe}} \tag{28}$$

where L is a linear operator to take out the position vector from the state vector $\boldsymbol{x}$, and $\boldsymbol{x}^k_{\text{obs}}$ is the position coordinates of the nearest ith obstacle within the detection radius of this agent.

3) Cost Function: Define the stage cost function as

$$\ell(\boldsymbol{x}_k, \boldsymbol{u}_k) = \|\boldsymbol{x}_k\|^2_{\boldsymbol{Q}} + \|\boldsymbol{u}_k\|^2_{\boldsymbol{R}} \tag{29}$$

where N is the prediction horizon, $\|\boldsymbol{x}_k\|^2_{\boldsymbol{Q}} = \boldsymbol{x}_k^T\boldsymbol{Q}\boldsymbol{x}_k$, $\|\boldsymbol{u}_k\|^2_{\boldsymbol{R}} = \boldsymbol{u}_k^T\boldsymbol{R}\boldsymbol{u}_k$, $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{R} \in \mathbb{R}^{m\times m}$ are weighting matrices, and $\boldsymbol{Q} \succ 0$ and $\boldsymbol{R} \succ 0$. The terminal cost is

$$\ell_f(\boldsymbol{x}_N) = \|\boldsymbol{x}_N\|^2_{\boldsymbol{P}} \tag{30}$$

where $\|\boldsymbol{x}_N\|^2_{\boldsymbol{P}} = \boldsymbol{x}_N^T\boldsymbol{P}\boldsymbol{x}_N$, $\boldsymbol{P} \in \mathbb{R}^{n\times n}$ is the penalty matrix, and $\boldsymbol{P} \succ 0$. Then, we have the cost function is

$$J(\boldsymbol{x}(t), U(t)) = \sum_{k=0}^{N-1} \ell\big(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}\big) + \ell_f\big(\boldsymbol{x}_{t+N|t}\big) \tag{31}$$

where $\boldsymbol{U}(t) = \begin{bmatrix}\boldsymbol{u}^T_{t|t} & \boldsymbol{u}^T_{t+1|t} & \cdots & \boldsymbol{u}^T_{t+N-1|t}\end{bmatrix}^T \in \mathbb{R}^{Nm}$ is the sequence of control inputs over the prediction horizon N.

Therefore, the following nonlinear MPC problem can be formulated as a constrained finite-horizon nonlinear quadratic optimal control problem at time t as

$$\begin{aligned}
\min_{\boldsymbol{U}} \ & J(\boldsymbol{x}(t), U(t)) \\
\text{s.t.} \ & \boldsymbol{x}_{t+k+1|t} = f(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}), k \in \mathbb{N}_{N-1} \\
& \boldsymbol{x}_{t|t} = \boldsymbol{x}(t) \\
& \boldsymbol{x}_{t+k|t} \in \mathcal{X}, \boldsymbol{u}_{t+k|t} \in \mathcal{U} \\
& \boldsymbol{\kappa}^T_{i,k}\big(\boldsymbol{x}_{t+k|t} - \Pi_{\mathcal{I}_{i,k}}\big(\boldsymbol{x}_{t+k|t}\big)\big) \\
& \quad \geq \sqrt{2\boldsymbol{\kappa}^T_{i,k}\boldsymbol{\Sigma}_{i,k}\boldsymbol{\kappa}_{i,k}}\,\text{erf}^{-1}(1-2\varphi) + d^2_{\text{safe}} \\
& \left\|L\boldsymbol{x}_{t+k|t} - L\boldsymbol{x}_{\text{obs}t+k}\right\| \geq d_{\text{safe}} \\
& \boldsymbol{x}_N \in \mathcal{X}_f
\end{aligned} \tag{32}$$

where the terminal constraint region $\mathcal{X}_f$ is a polytope.

D. Stability Analysis

Ahead of the stability analysis suitably characterizing the performance of the proposed methodology, the following assumptions, definitions, and lemmas are introduced to provide a sufficient condition to prove the stability. Here, we focus on the conditions for the uniform asymptotical stability of the origin of system (26) under the cost function (31), physical limitation constraint (27), and terminal region $\mathcal{X}_f$, with the feedback control law.

Assumption 1: Consider the problem (32). If the initial problem at time 0 is feasible, then the problem at time t is feasible for all $t > 0$. (It may be noted that this is a reasonable assumption pertaining to the posed physical system being a typical actual system where actual real-world solutions can be admissible. This should be expected of typical actual systems.)

Definition 1: A continuous function $V(t, \boldsymbol{x})$ is a locally positive-definite function, if $V(t, 0) = 0$ and

$$V(t, \boldsymbol{x}) \geq \gamma(|\boldsymbol{x}|) \quad \forall \boldsymbol{x} \in \mathbb{B}_d, \quad t \geq 0 \tag{33}$$

where $\mathbb{B}_d$ is a ball centered in the origin with radius d, and the function $\gamma(\cdot)$ is continuous and strictly increasing with $\gamma(0) = 0$.

Definition 2: A function $V(t, \boldsymbol{x})$ is decrescent with $\boldsymbol{x} \in \mathcal{X}$, if there exists a function $\epsilon(\cdot)$ such that $V(t, \boldsymbol{x}(t)) \leq \epsilon(|\boldsymbol{x}|)$, where the function $\epsilon(\cdot)$ is continuous and strictly increasing with $\epsilon(0) = 0$.

Lemma 5: Consider the system (26) with $L\boldsymbol{x}(t) = \boldsymbol{p}(t)$, and let a continuous function $V_N(t, \boldsymbol{x}(t))$ be stated as the value function. The origin of the system is locally uniformly asymptotically stable, if $V_N(t, \boldsymbol{x}(t))$ is a local positive-definite function.

Proof: The proof of this result stated above is shown in [29]. ∎

Lemma 6: For the system (26), with the stage cost function $\ell(\cdot)$ and the terminal cost function $\ell_f(\cdot)$, the value function $V_N(t, \boldsymbol{x}(t))$ is a decrescent function in the domain $\mathcal{X}$ for $\boldsymbol{x}(t) \in \mathcal{X}$.

Proof: The value function is

$$V_N(t, \boldsymbol{x}(t)) = \sum_{k=0}^{N-1}\Big(\left\|\boldsymbol{x}_{t+k|t}\right\|^2_{\boldsymbol{Q}} + \left\|\boldsymbol{u}_{t+k|t}\right\|^2_{\boldsymbol{R}}\Big) + \left\|\boldsymbol{x}_{t+N|t}\right\|^2_{\boldsymbol{P}} \tag{34}$$

which can be rewritten as

$$V_N(t, \boldsymbol{x}(t)) = \begin{bmatrix}\boldsymbol{x}^T(t) & \boldsymbol{U}^T(t)\end{bmatrix}\boldsymbol{\Xi}\begin{bmatrix}\boldsymbol{x}(t) \\ \boldsymbol{U}(t)\end{bmatrix} \tag{35}$$

where the matrix $\boldsymbol{\Xi} \in \mathbb{R}^{(n+Nm)\times(n+Nm)}$ can be determined by $\boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{P}, \boldsymbol{A}_t$, and $\boldsymbol{B}_t$. The components of $\boldsymbol{u}(t)$ are bounded since it is a solution of problem (32) and the set $\mathcal{U}$ is compact. The same property can also apply to $\boldsymbol{x}(t)$. The partial derivative of $f(\boldsymbol{x}, \boldsymbol{u})$ with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$ is bounded and, thus, matrices $\boldsymbol{A}_t$ and $\boldsymbol{B}_t$ are bounded. Therefore, $V_N(\cdot, \cdot)$ is

bounded and we can always find a positive-definite function $\epsilon(\cdot)$ such that $V_N(t, \boldsymbol{x}(t)) \leq \epsilon(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathcal{X}$ [30]. According to Definition 2, the value function $V_N(t, \boldsymbol{x}(t))$ is a decrescent function in the domain $\mathcal{X}$ for $\boldsymbol{x}(t) \in \mathcal{X}$. This completes the proof of the result. ■

In the following, we use $V_N(t, \boldsymbol{x}(t))$ as a Lyapunov function to find a sufficient condition to prove the stability.

Theorem 2: Consider the system (26) with the terminal constraint $\mathcal{X}_f = 0$, physical limitation constraint (27), and the cost function (31). Let $\Delta V_N(t, \boldsymbol{x}(t)) = V_N(t-1, \boldsymbol{x}(t-1)) - V_N(t, \boldsymbol{x}(t))$, then the function $\Delta V_N(\cdot, \boldsymbol{x}(\cdot))$ is a locally positive-definite function if

$$\ell\big(\boldsymbol{x}_{t+N-1|t}, \boldsymbol{u}_{t+N-1|t}\big) < \ell\big(\boldsymbol{x}_{t-1|t-1}, \boldsymbol{u}_{t-1|t-1}\big) - \sum_{i=0}^{N-2} \big\| \boldsymbol{x}_{t+i|t} - \boldsymbol{x}_{t+i|t-1}\big)\big\|_{\boldsymbol{Q}}^2. \quad (36)$$

Then, it follows that the origin of the closed-loop system is uniformly, locally asymptotically stable.

Proof: Recall that the value function at time t is

$$V_N(t, \boldsymbol{x}(t)) = \sum_{k=0}^{N-1} \Big(\big\| \boldsymbol{x}_{t+k|t} \big\|_{\boldsymbol{Q}}^2 + \big\| \boldsymbol{u}_{t+k|t} \big\|_{\boldsymbol{R}}^2 \Big) + \Big\| \boldsymbol{x}_{t+N|t}^T \Big\|_{\boldsymbol{P}}^2 \quad (37)$$

where $\boldsymbol{u}_{t+k,t}$ with $k = 0, 1, \ldots, N-1$ is the solution of the optimization problem at time t. The state of the system at time t is $\boldsymbol{x}(t) = f(\boldsymbol{x}(t-1), u_{t-1|t-1})$. At time $t-1$, we have the solution $\boldsymbol{U}_{t-1} = \begin{bmatrix} u_{t-1|t-1}^T & u_{t|t-1}^T & \cdots & u_{t+N-2|t-1}^T \end{bmatrix}^T$. Set the sequence $\hat{\boldsymbol{U}}_t = \begin{bmatrix} u_{t|t-1}^T & u_{t+1|t-1}^T & \cdots & u_{t+N-2|t-1}^T & u_{t+N-1|t}^T \end{bmatrix}^T$ which is feasible for the problem, and is obtained from $\boldsymbol{U}_{t-1}$ by removing $u_{t-1|t-1}$ and adding $u_{t+N-1|t}$. The function $\Delta V_N(t, \boldsymbol{x}(t))$ can be rewritten as

$$\begin{aligned}
\Delta V_N(t, \boldsymbol{x}(t)) &= \sum_{k=0}^{N-1} \big\| \boldsymbol{x}_{t+k-1|t-1} \big\|_{\boldsymbol{Q}}^2 + \sum_{k=0}^{N-1} \big\| \boldsymbol{u}_{t+k-1|t-1} \big\|_{\boldsymbol{R}}^2 \\
&\quad + \big\| \boldsymbol{x}_{t+N-1|t-1} \big\|_{\boldsymbol{P}}^2 - \sum_{k=0}^{N-1} \big\| \boldsymbol{x}_{t+k|t} \big\|_{\boldsymbol{Q}}^2 \\
&\quad - \sum_{k=1}^{N-1} \big\| \boldsymbol{u}_{t+k-1|t-1} \big\|_{\boldsymbol{R}}^2 - \big\| \boldsymbol{u}_{t+N-1|t} \big\|_{\boldsymbol{R}}^2 \\
&\quad - \big\| \boldsymbol{x}_{t+N|t} \big\|_{\boldsymbol{P}}^2 \\
&= \sum_{k=0}^{N-2} \Big(\big\| \boldsymbol{x}_{t+k|t-1} \big\|_{\boldsymbol{Q}}^2 - \big\| \boldsymbol{x}_{t+k|t} \big\|_{\boldsymbol{Q}}^2 \Big) + \big\| \boldsymbol{x}_{t-1|t-1} \big\|_{\boldsymbol{Q}}^2 \\
&\quad + \big\| \boldsymbol{u}_{t-1|t-1} \big\|_{\boldsymbol{R}}^2 - \big\| \boldsymbol{x}_{t+N-1|t} \big\|_{\boldsymbol{Q}}^2 - \big\| \boldsymbol{u}_{t+N-1|t} \big\|_{\boldsymbol{R}}^2 \\
&\geq - \sum_{k=0}^{N-2} \big\| \boldsymbol{x}_{t+k|t} - \boldsymbol{x}_{t+k|t-1} \big\|_{\boldsymbol{Q}}^2 \\
&\quad - \ell\big(\boldsymbol{x}_{t+N-1|t}, \boldsymbol{u}_{t+N-1|t}\big) \\
&\quad + \ell\big(\boldsymbol{x}_{t-1|t-1}, \boldsymbol{u}_{t-1|t-1}\big). \quad (38)
\end{aligned}$$

If the condition (36) holds, $\Delta V_N(t, \boldsymbol{x}(t))$ is a locally positive-definite function since the right-hand side of (38) is positive and bounded. Then, the function $\Delta V_N(t, \boldsymbol{x}(t))$ is a locally positive-definite function and it is a decrescent function based on Lemma 6. According to Lemma 5, the origin of the system is uniformly asymptotically stable. ■

Remark 4: The condition (36) is established for the stated nonlinear system above, and it directly leads to an additional convex constraint to be embedded and incorporated in the required MPC design and algorithm. With all of these in place at this point, this then is an appropriate applicable stability result for the methodology proposed here.

Algorithm 1 Numerical Procedures for Nonlinear MPC Problem at Time t With Chance Constraints

1: Initialize state vector of the agent $\boldsymbol{x}_0$; initialize the number of repeats $n_{\text{rep}} = 1$; set both of the maximum number of repeats n_{set} and the maximum computation time $t_{\max}$ to the reasonable values.
2: Compute the initial value of the cost function by (31).
3: **for** $k = 0, 1, \cdots, N-1$ **do**
4: Compute $\boldsymbol{x}_{t+k+1|t}$ based on $\boldsymbol{x}_{t+k|t}$ using (26).
5: Generate the constraints of the physical limitations by (27).
6: Generate the constraints of collision avoidance to moving obstacles by (19).
7: Generate the constraints of collision avoidance to static obstacles by (28).
8: Generate the additional constraint based on the stability condition (36).
9: **end for**
10: Compute the cost function by (31).
11: Solve the nonlinear MPC problem (32).
12: **if** There exists a solution **then**
13: **return** control sequence $\boldsymbol{U}(t)$
14: **else**
15: Add slack vectors on these constraints and cost function.
16: $n_{\text{rep}} = n_{\text{rep}} + 1$.
17: Go to line 11.
18: **end if**
19: Compute the computation time collapse t_{comp}.
20: **if** $n_{\text{rep}} > n_{\text{set}}$ or $t_{\text{comp}} > t_{\max}$ **then**
21: Turn on the backup controller.
22: **end if**

E. Proposed Algorithm

At time t, the cost function is optimized under the constraints in (32) to obtain the optimal control sequence $\boldsymbol{U}(t)$. It is worthwhile to mention that only the first control input $\boldsymbol{u}_{t|t}$ will be executed. The multiple shooting method [31] is used to solve this nonlinear optimization problem with multiple constraints (32). First, discretize the system dynamics and constraints at each time t over a coarse discrete-time grid $k = 0, 1, \ldots, N$ with sampling step Δt. For each time t, a boundary value problem is solved by imposing some additional continuity constraints. This problem can be expressed as a nonlinear program which can be solved using sequential quadratic programming. The interior-point method or the

active set method can be applied to solve the corresponding quadratic program.

Here, Algorithm 1 is used to obtain the optimal control inputs at time t and, thus, the optimal control sequence can be obtained. If there is no solution of this problem but solving time t_{comp} is less than the predefined maximum computation time limits $t_{\max}$, a non-negative slack vector $\Theta_k \in \mathbb{R}^{n_x}$ will be added to soften the inequality constraints [32], as shown in

$$\begin{aligned} \boldsymbol{H}_x \boldsymbol{x}_{t+k|t} &\leq \boldsymbol{h}_x + \Theta_k \\ \Theta_k &\geq 0 \end{aligned} \tag{39}$$

where the symbol $\geq$ means element-wisely no less than 0, the subscript $\cdot_k$ denotes each step in prediction horizon of the MPC problem with $k \in \mathbb{N}_{N-1}$, and $\boldsymbol{H}_x$ and $\boldsymbol{h}_x$ are arbitrary given matrix and vector. The value of the slack vector relies on the degree of associated acceptable violation of the constraints.

The slack vector $\tilde{\Theta}_k$ can also be added in the equality constraints to transform the equality constraints into tube-like constraints, as shown in

$$\begin{aligned} f\left(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}\right) - \tilde{\Theta}_k \leq \boldsymbol{x}_{t+k+1|t} \leq f\left(\boldsymbol{x}_{t+k|t}, \boldsymbol{u}_{t+k|t}\right) + \tilde{\Theta}_k \\ \tilde{\Theta}_k \geq 0. \end{aligned} \tag{40}$$

Moreover, the slack vector $\hat{\Theta}_k$ can also be added to the cost function as a scalar weight to ensure that the slacking is not abused [32], as shown in

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \ & \sum_{k=0}^{N-1} \left(\left\| \boldsymbol{x}_{t+k|t} \right\|_{\boldsymbol{Q}}^2 + \left\| \boldsymbol{u}_{t+k|t} \right\|_{\boldsymbol{R}}^2 + \rho \hat{\Theta}_k^T \hat{\Theta}_k \right) \\ & + \left\| \boldsymbol{x}_{t+N|t} \right\|_{\boldsymbol{P}}^2 \end{aligned} \tag{41}$$

where ρ is the penalty of the slack vector.

Remark 5: Adding slacking vectors on control input constraints is not reasonable, as the inputs often originate from an actuator which has hard limits constraining the force, torque, etc.

If there is no feasible solution or the computation time to solve this problem t_{comp} exceeds the maximum computation time limits $t_{\max}$, a backup controller will be invoked, thereby continuing the control progress. For example, a conservatively tuned PID controller can be used which sacrifices performance for relaxed constraints satisfaction. An alternative way is to execute the control inputs at the upper/lower bound.

IV. Case Study

Among all of the applications of trajectory generation, a case study on a UAV system would certainly be an ideal test platform for the 3-D trajectory generation problem, since trajectory planning typically needs to work in the 3-D state space with multiple degrees of freedom and multiple constraints due to its dynamical characteristics and physical limits in a typical UAV application. To demonstrate the effectiveness of our proposed method, a UAV (quadcopter) system is used as an application test platform for trajectory generation in this section.

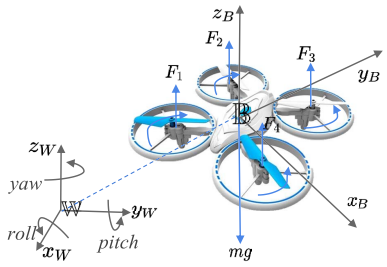


Fig. 1. Model of the quadcopter.

A. Plant Model

The world coordinate system $\mathbb{W}$ and the robot body coordinate system $\mathbb{B}$ are shown in Fig. 1, where x_W, y_W, and z_W are three dimensions in the world-fixed frame and x_B, y_B, and z_B are in the body-fixed frame. As shown in Fig. 1, each quadcopter is equipped four rotors. For each rotor, there are a vertical force due to the rotation of the rotor and a moment perpendicular to the plane of the propeller rotation. Therefore, there are four vertical forces F_1, F_2, F_3, and F_4 to overcome the gravity and drive the quadcopter. The dynamic model of a quadcopter can be represented by (42a) with neglecting the aerodynamic and gyroscopic effects [33]

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{R}(\phi, \theta, \psi)^T \boldsymbol{v}(t) \tag{42a}$$

$$\dot{\boldsymbol{v}}(t) = -\boldsymbol{\omega}(t) \times \boldsymbol{v}(t) + g\boldsymbol{R}(\phi, \theta, \psi)\boldsymbol{e} + \boldsymbol{e}T/m \tag{42b}$$

$$\dot{\boldsymbol{\zeta}}(t) = \boldsymbol{W}(\phi, \theta, \psi)\boldsymbol{\omega}(t) \tag{42c}$$

$$\dot{\boldsymbol{\omega}}(t) = \boldsymbol{J}^{-1}(-\boldsymbol{\omega}(t) \times \boldsymbol{J}\boldsymbol{\omega}(t) + \boldsymbol{\tau}). \tag{42d}$$

Here, (42a) models the position of quadcopter in the world coordinates $\boldsymbol{p} = \begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^T \in \mathbb{R}^3$; (42b) represents the velocity of the quadcopter in three dimensions $\boldsymbol{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T \in \mathbb{R}^3$; $\boldsymbol{\zeta} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \in \mathbb{R}^3$ denotes the roll, pitch, and yaw angle, respectively; the angular velocity in three dimensions is represented by $\boldsymbol{\omega} = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \in \mathbb{R}^3$; $\boldsymbol{e} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \in \mathbb{R}^3$; $\boldsymbol{\tau} = \begin{bmatrix} \tau_x & \tau_y & \tau_z \end{bmatrix}^T \in \mathbb{R}^3$ represents the torques of the quadcopter in each dimension; g is the gravitational acceleration; m is the mass of this quadcopter; T denotes the total thrust; $\boldsymbol{J} = \text{diag}(J_x, J_y, J_z) \in \mathbb{R}^{3\times 3}$ denotes the moment of inertia of the quadcopter; and $\boldsymbol{R}(\phi, \theta, \psi)$ and $\boldsymbol{W}(\phi, \theta, \psi)$ denote the rotation matrices of the quadcopter (see [33]).

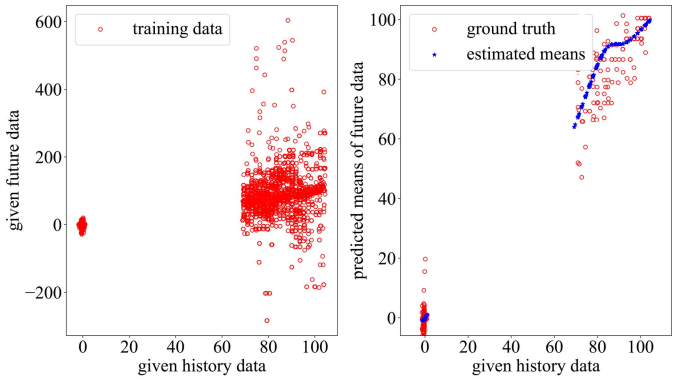
The rotor thrusts of the four rotors are chosen as control inputs, that is, $\hat{\boldsymbol{u}} = \begin{bmatrix} F_1 & F_2 & F_3 & F_4 \end{bmatrix}^T \in \mathbb{R}^4$, and then we have the relationship between individual thrusts and individual torques which is expressed by

$$\begin{bmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} -1 & -1 & -1 & -1 \\ 0 & -L & 0 & L \\ L & 0 & -L & 0 \\ -c & c & -c & c \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{bmatrix} \tag{43}$$

where L is the distance from the rotor to the center of gravity of the quadrotor and c is a constant that relates the rotor angular momentum to the rotor thrust.

Define the state vector as

$$\begin{aligned} \boldsymbol{x} &= \begin{bmatrix} p_x & p_y & p_z & v_x & v_y & v_z & \phi & \theta & \psi & \omega_x & \omega_y & \omega_z \end{bmatrix}^T \\ &= \begin{bmatrix} \boldsymbol{p}^T & \boldsymbol{v}^T & \boldsymbol{\zeta}^T & \boldsymbol{\omega}^T \end{bmatrix}^T \in \mathbb{R}^{12}. \end{aligned} \tag{44}$$



(a) (b)

Fig. 2. Training and testing results of trajectory prediction. (a) Results of the training process. (b) Results of the testing process.

The dynamic model of a quadcopter can be formulated by the form of $\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + \tilde{\boldsymbol{B}}\hat{\boldsymbol{u}}$. Let $\hat{\boldsymbol{u}} = \boldsymbol{u}_{\text{eq}} + \boldsymbol{u}$ with $\hat{\boldsymbol{u}} \in \mathbb{R}^4$, where $\boldsymbol{u}_{\text{eq}} = \left[\frac{mg}{4} \quad \frac{mg}{4} \quad \frac{mg}{4} \quad \frac{mg}{4}\right]^T$ is used to overcome the gravity of the quadcopter. Therefore, the quadcopter system can be represented by

$$\begin{bmatrix} \dot{\boldsymbol{p}} \\ \dot{\boldsymbol{v}} \\ \dot{\boldsymbol{\zeta}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}(\phi,\theta,\psi)^T\boldsymbol{v} \\ -\boldsymbol{\omega}\times\boldsymbol{v} + g\boldsymbol{R}(\phi,\theta,\psi)\boldsymbol{e} \\ \boldsymbol{W}(\phi,\theta,\psi)\boldsymbol{\omega} \\ \boldsymbol{J}^{-1}(-\boldsymbol{\omega}\times\boldsymbol{J}\boldsymbol{\omega}) \end{bmatrix} + \tilde{\boldsymbol{B}}\big(\boldsymbol{u}_{\text{eq}} + \boldsymbol{u}\big). \quad (45)$$

Due to space constraints, more details of the matrix $\tilde{\boldsymbol{B}} \in \mathbb{R}^{12\times 4}$ refer to [33].

B. Model Linearization

The expression (45) is nonlinear and time variant, so the state-dependent coefficient factorization [34] is used to handle and address the nonlinear dynamics. The resulting state-space expression can be expressed by

$$\boldsymbol{x}_{k+1|t} = \Big(\tilde{\boldsymbol{A}}_t\Delta t + I\Big)\boldsymbol{x}_{t+k|t} + \tilde{\boldsymbol{B}}\Delta t\big(\boldsymbol{u}_{\text{eq}} + \boldsymbol{u}_{t+k|t}\big) \quad (46)$$

where Δt is the sampling time interval. Since $\tilde{\boldsymbol{A}}_t$ and $\tilde{\boldsymbol{B}}$ are dependent on the current state $\boldsymbol{x}$, this state-space representation is a pseudolinear form, and then we can suitably consider the system matrices to be constant during the prediction horizon. The full details of the matrix $\tilde{\boldsymbol{A}}_t \in \mathbb{R}^{12}$ are shown in Appendix A. The control design focuses on $\boldsymbol{u}_{t+k|t}$. When $\boldsymbol{u}_{t+k|t} = 0$, the quadcopter lies in an equilibrium situation as $\tilde{\boldsymbol{A}}_t\boldsymbol{x}_{t+k|t} + \tilde{\boldsymbol{B}}\boldsymbol{u}_{\text{eq}} = 0$.

C. Problem Formulation

According to the aforementioned analysis, a nonlinear MPC problem with chance constraints at time t can be formulated as

$$\begin{aligned} \min_{\boldsymbol{u}} \ & \sum_{k=0}^{N}\left\|\boldsymbol{x}_{t+k|t} - \boldsymbol{x}_{\text{ref}t+k}\right\|_{\boldsymbol{Q}} + \sum_{k=0}^{N-1}\left\|\boldsymbol{u}_{t+k|t}\right\|_{\boldsymbol{R}} \\ \text{s.t.} \ & \boldsymbol{x}_{k+1|t} = \Big(\tilde{\boldsymbol{A}}_t\Delta t + I\Big)\boldsymbol{x}_{t+k|t} + \tilde{\boldsymbol{B}}\Delta t\big(\boldsymbol{u}_{\text{eq}} + \boldsymbol{u}_{t+k|t}\big) \\ & \left\|\boldsymbol{p}_{t+k+1|t} - \boldsymbol{p}_{\text{obs},j}\right\| \geq d_{\text{safe}} \\ & -\pi \leq \boldsymbol{\phi}, \boldsymbol{\psi} \leq \pi, \ -\frac{\pi}{2} \leq \boldsymbol{\theta} \leq \frac{\pi}{2} \\ & \underline{\boldsymbol{v}} \leq \boldsymbol{v}_{t+k+1|t} \leq \overline{\boldsymbol{v}} \\ & \underline{\boldsymbol{u}} \leq \boldsymbol{u}_{t+k|t} \leq \overline{\boldsymbol{u}} \\ & \boldsymbol{\kappa}_{i,k+1}^T\big(\boldsymbol{p}_{t+k+1|t} - \Pi_{\mathcal{I}_{i,k+1}}\big(\boldsymbol{p}_{t+k+1|t}\big)\big) \\ & \geq \sqrt{2\boldsymbol{\kappa}_{i,k+1}^T\boldsymbol{\Sigma}_{i,k+1}\boldsymbol{\kappa}_{i,k+1}}\,\text{erf}^{-1}(1-2\varphi) + d_{\text{safe}}^2 \\ & \left\|\boldsymbol{x}_{t+N-1|t} - \boldsymbol{x}_{\text{ref}t+N-1}\right\|_{\boldsymbol{Q}}^2 + \left\|\boldsymbol{u}_{t+N-1|t}\right\|_{\boldsymbol{R}}^2 \leq \varpi \\ & \forall j \in \mathbb{N}_{n_s} \cup \mathbb{N}_{n_o}, \quad \forall i \in \mathbb{N}_{n_o} \end{aligned} \quad (47)$$

where $\boldsymbol{x}_{\text{ref}t+k}$ denotes the reference states at step $t+k$ which can be planned by some path-planning algorithms [35], such as A*, D* Lite, HLT*, RRT, etc., $\boldsymbol{p}_{\text{obs},j}$ is the position vector of static obstacles, the set $\mathbb{N}_{n_s} = \{1, 2, \ldots, n_s\}$ and n_s is the number of the ith static obstacle, $\underline{\boldsymbol{v}}, \underline{\boldsymbol{u}}$ and $\overline{\boldsymbol{v}}, \overline{\boldsymbol{u}}$ are the lower and upper bounds of velocity and control input limitations for this UAV, and ϖ is given by

$$\varpi = \left\|\boldsymbol{x}_{t-1|t-1} - \boldsymbol{x}_{\text{ref}t-1}\right\|_{\boldsymbol{Q}}^2 + \left\|\boldsymbol{u}_{t-1|t}\right\|_{\boldsymbol{R}}^2 - \sum_{i=0}^{N-2}\left\|\boldsymbol{x}_{t+i|t} - \boldsymbol{x}_{t+i|t-1}\right\|_{\boldsymbol{Q}}^2 - e \quad (48)$$

with $e > 0$ [36]. Remarkably, the last constraint of this problem (47) represents the stability condition (36).

D. Simulation Results

Parameters of this quadcopter are shown in Table I. All of the simulations are implemented in Python 3.7 environment on a PC with Intel i5 CPU@3.30 GHz.

1) Inference Results of Trajectory Prediction: The input feature of vBGMM is the Chebyshev coefficients of the trajectories in the past and future. The dataset includes 1000 planned trajectories with 3-D positions $(\boldsymbol{p}_x, \boldsymbol{p}_y, \boldsymbol{p}_z)$ in the 3-D clustered environment with multiple static obstacles. The Chebyshev approximation is used in all these trajectories to obtain the input data in the feature space, that is, coefficients of this approximation $\boldsymbol{a}_x, \boldsymbol{a}_y, \boldsymbol{a}_z$. We split these trajectories with history partition with a length of 70% and future partition with a length of 40%. There is a length of 10% overlapping segment between history and future trajectories snippets. The total dataset is divided into a training set and a test set with 875 and 125 trajectories. Experimental results show that the Chebyshev approximation performance is better when the degree of freedom of this approximation is set to 4, which results in the feature space with a dimension of $D = 15$. The initial number of mixing components K is set to 30. The parameters of the prior distribution for the vBGMM are $\beta_0 = 1$, $\alpha_0 = 1$, and $\nu_0 = 5$, and the initial mean vector $\mathbf{m}_0$ and precision matrix $\mathbf{W}_0$ are given by the mean of training data $\boldsymbol{\mu}(X)$ and covariance of training data $\boldsymbol{\Sigma}(X)$. Here, X means the training data. The allowable probability threshold of collision φ is given by 0.05.

The training and testing results are shown in Fig. 2 with the root-mean-square (RMS) deviation of 1.681. The small RMS error indicates that the vBGMM can effectively learn and infer the predictive posterior distribution.

The variational lower bound can be used to monitor the convergence and check the correctness of the variational inference process. The maximization of the variational lower bound indicates a good estimation of the posterior distribution. At each

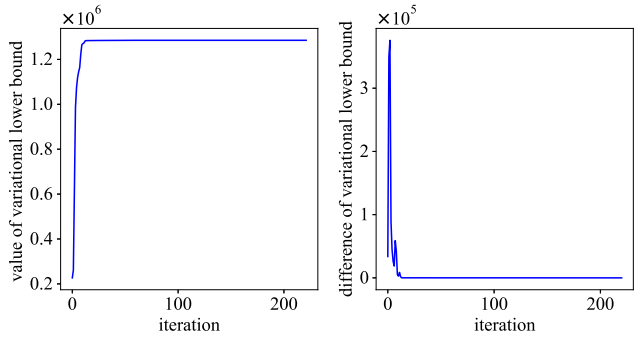


Fig. 3. Change of variational lower bound and difference of variational lower bound during iterations.

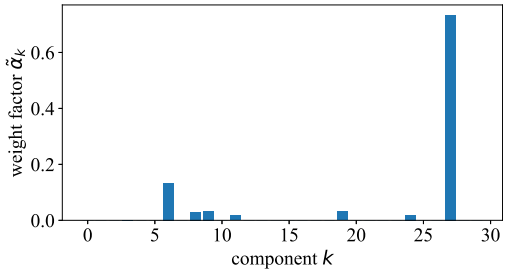


Fig. 4. Weighting factors in each component after training.

step of the iterative re-estimation process, the lower bound will not decrease. Our prediction result of the variational lower bound is shown in Fig. 3. The stopping criterion in terms of the variational lower bound difference is set to 10^{-12}.

The number of mixture components K can be automatically decreasing due to the sparsity property of variational approximation. The sparsity performance can be illustrated by the value of weighting factor $\tilde{\alpha}_k$, as shown in Fig. 4. According to Fig. 4, the number dominant components decreases from the initial value of 30 to 7.

Here, we just take one predicted trajectory as an example to show the predictive performance. Figs. 5 and 6 show the prediction trajectory which consists of means of the predicted position with uncertainties.

2) Results of the Nonlinear MPC Solution: In this simulation, there are ten static random obstacles and three moving obstacles, as shown in Fig. 7. According to this figure, such a control method can track the reference trajectory with small tracking errors except for collision avoidance. When tracking the reference trajectory, avoiding the obstacles based on the prediction should also be satisfied for the UAV. In this simulation, the computation time limit $t_{\max}$ is set to 0.2 s. The average of solving time t_{comp} in each sampling time t during the whole running process is about 0.093 s.

In Fig. 8, the perturbations of positions represent the larger tracking errors in three dimensions x, y, and z, which illustrates the collision-avoidance behaviors of the UAV to the static obstacles and moving obstacles. Compared with the results of nonlinear MPC controller without the probabilistic prediction (magenta line), the tracking errors due to avoidance of moving obstacles with probabilistic prediction are smaller and the time when the UAV test platform starts to

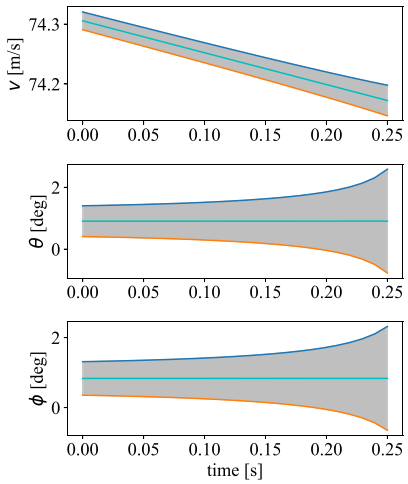


Fig. 5. Mean and uncertainties of one predictive states in each of spherical coordinates over prediction horizon (cyan line: means in each spherical coordinate with time; gray region: uncertainties in each spherical coordinate with time; orange line: lower bound of the uncertainties in each spherical coordinate with time; and blue line: upper bound of the uncertainties in each spherical coordinate with time).

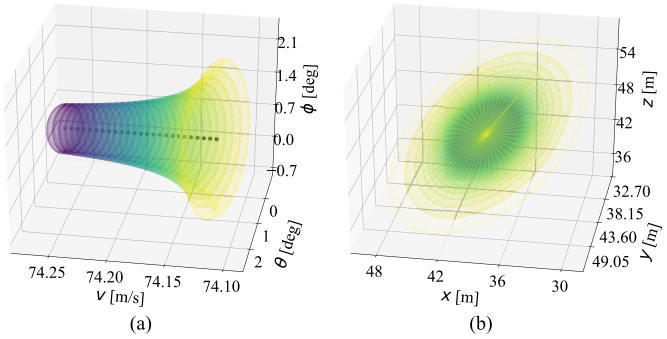


Fig. 6. Means and uncertainties of one predictive trajectory in spherical and Cartesian coordinates over a prediction horizon. (a) Means and uncertainties in the spherical coordinates. (b) Means and uncertainties in the Cartesian coordinates. (Black dots: means with time; and colorful ellipsoids: uncertainties with time).

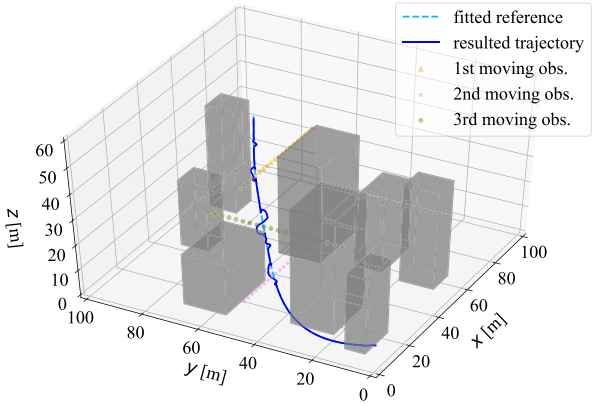


Fig. 7. Trajectories generated in the 3-D environment with probabilistic prediction.

avoid the moving obstacles is certainly earlier. This indicates that the prediction process can effectively help to avoid the collision more accurately and prepare to avoid potential collision measurably in advance.

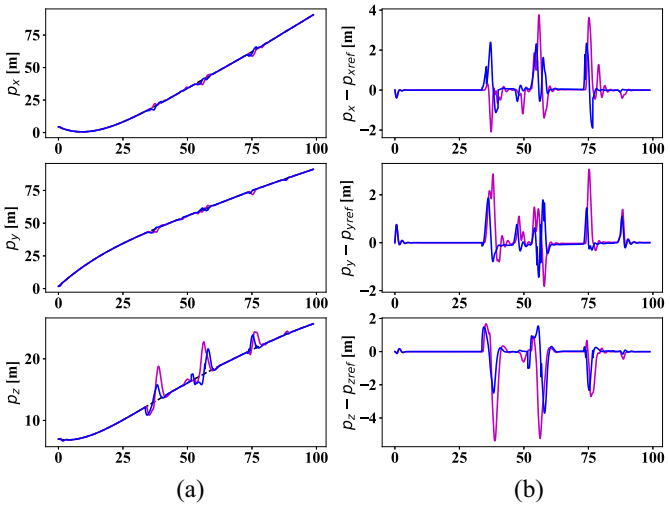


Fig. 8. Comparison of positions and tracking errors in three dimensions with probabilistic prediction and without prediction. (a) Comparison of positions. (b) Comparison of tracking errors. (Blue line: with prediction; magenta line: without prediction]; and dark dashed line: reference trajectory).

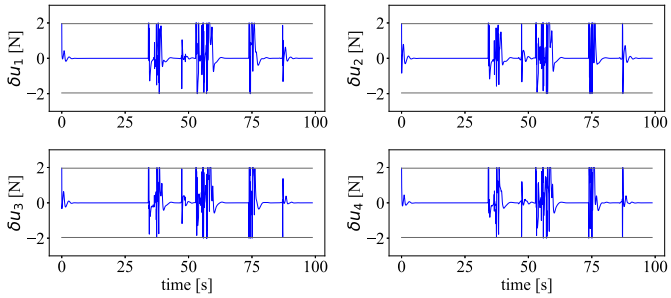


Fig. 9. Control inputs with probabilistic prediction.

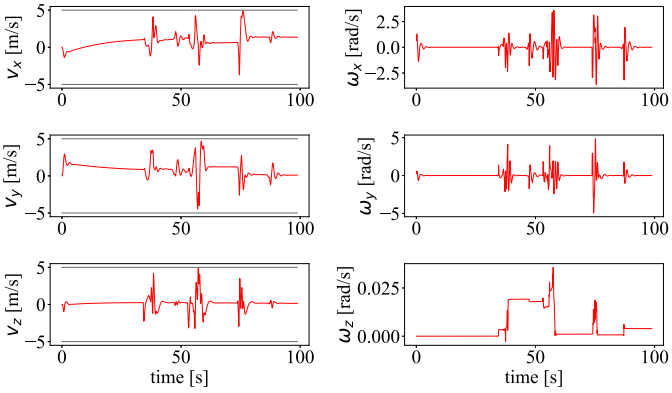


Fig. 10. Velocities and angular velocities in three dimensions with probabilistic prediction.

During generating the trajectories, the constraints of control inputs should be satisfied, as shown in Fig. 9. In this figure, all control inputs are constrained in the predefined bounded range $[-2, 2]$ N. Besides, constraints of velocity also need to be satisfied. In Fig. 10, it is obvious that the velocity in each dimension x, y, z is successfully confined into the given range $[-5, 5]$ m/s and the angular velocities in three dimensions are illustrated in the right column of this figure.

As aforementioned, the shortest distance between the current position of the host UAV and the nearest static obstacle

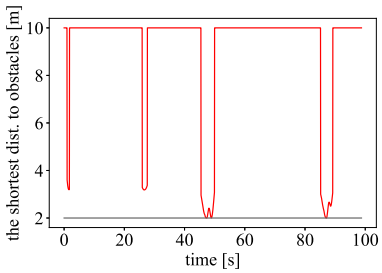


Fig. 11. Shortest distance to the nearest static obstacle with probabilistic prediction.

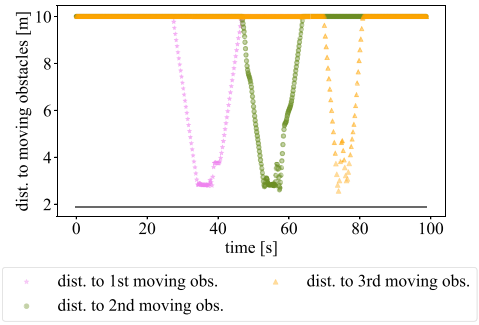


Fig. 12. Distance to the three moving obstacles with probabilistic prediction.

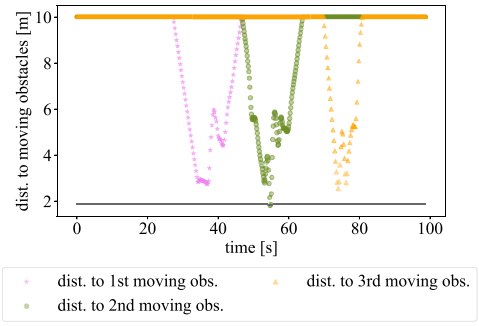


Fig. 13. Distance to the three moving obstacles without probabilistic prediction.

should be no less than the predefined $d_{\text{safe}} = 2$ m, as shown in Fig. 11. Figs. 12 and 13 show the distance from the robot to the three moving obstacles when doing probabilistic prediction by vBGMM and not doing prediction, respectively. The pink, green, and yellow dotted lines represent the distance to the first, second, and third moving obstacles. Note that if the distance to the moving obstacles is greater than 10 m, the distance will cap at 10 m. Compared with the results of nonlinear MPC controller without prediction in Fig. 13, it can be observed that the risk of future collision will be higher (without prediction). If there are some fast-moving obstacles, in the methodology without prediction, the probability of collision will be much higher; while the nonlinear MPC method with prediction can foresee the future probabilistic trajectory and avoid the obstacles as soon as it is detected by the UAV, instead of encountering violation of the shortest safety distance condition (without prediction).

TABLE I
PARAMETER SETTINGS FOR THE UAV DYNAMIC MODEL

Definition	Notation	Value	Unit
Mass	m	0.8	kg
Gravity acceleration	g	9.81	m/s^2
Moment of inertia in x dim.	J_x	0.0244	kg$\cdot$ m^2
Moment of inertia in y dim.	J_y	0.0244	kg$\cdot$ m^2
Moment of inertia in z dim.	J_z	0.0436	kg$\cdot$m^2
Distance from center to rotor	L	0.162	m
Ratio of rotor angular momentum to lift	γ	2.17×10^{-3}	m
Sampling time interval	Δt	0.05	s
State weighting matrix	$\boldsymbol{Q}$	I_{12}	-
Input weighting matrix	$\boldsymbol{R}$	I_4	-
Prediction horizon	N	25	-
Control input difference upper/lower bound	$\delta \boldsymbol{u}_{\max}$, $\delta \boldsymbol{u}_{\min}$	1.96, -1.96	N
Velocity upper/lower bound	$\boldsymbol{v}_{\max}$, $\boldsymbol{v}_{\min}$	5, -5	m/s
The safety distance to obstacles	d_{safe}	2	m
Detection radius	r_{det}	10	m

V. CONCLUSION

In this article, a suitably interesting concept of a chance-constrained nonlinear MPC approach with probabilistic prediction is proposed to generate trajectories for appropriate agents in a cluttered and unknown environment, and also in the presence of the parametric uncertainty and sensor noise. Variational inference is used to infer the parameters of the prediction distribution of future trajectory; and chance constraints are formulated based on the prediction results and reformulated as deterministic constraints by enlarging the ellipsoid collision region into half space. Then, this nonlinear MPC problem embedded with collision avoidance chance constraints is designed and solved iteratively by an optimization approach. Simulation results on a rather appropriate test case of a quadcopter system show that our formulation of this nonlinear MPC method (integrated with chance constraints based on the probabilistic prediction) can very effectively reduce the risk of potential future collision and avoid the obstacles accurately while meeting all of the other typical environmental and physical constraints.

APPENDIX A
MATRIX IN THE QUADCOPTER MODEL

Matrix $\tilde{\boldsymbol{A}}_t$ in (46) is given by

$$\tilde{\boldsymbol{A}}_t = \begin{bmatrix} \boldsymbol{0}_{(3,3)} & \boldsymbol{R}_{(3,3)}^T & \boldsymbol{0}_{(3,3)} & \boldsymbol{0}_{(3,3)} \\ \boldsymbol{0}_{(3,3)} & \boldsymbol{A}_{(3,3)} & \boldsymbol{B}_{(3,3)} & \boldsymbol{C}_{(3,3)} \\ \boldsymbol{0}_{(3,3)} & \boldsymbol{0}_{(3,3)} & \boldsymbol{0}_{(3,3)} & \boldsymbol{W}_{(3,3)} \\ \boldsymbol{0}_{(3,3)} & \boldsymbol{0}_{(3,3)} & \boldsymbol{0}_{(3,3)} & \boldsymbol{D}_{(3,3)} \end{bmatrix}$$

where $\boldsymbol{R}_{(3,3)} = \boldsymbol{R}(\phi, \theta, \psi)$, $\boldsymbol{W}_{(3,3)} = \boldsymbol{W}(\phi, \theta, \psi)$, and $\boldsymbol{0}_{(3,3)}$ is a 3-by-3 zero matrix. The block matrices $\boldsymbol{A}_{(3,3)}$, $\boldsymbol{B}_{(3,3)}$, $\boldsymbol{C}_{(3,3)}$, and $\boldsymbol{D}_{(3,3)}$ in $\tilde{\boldsymbol{A}}_t$ are shown as

$$\boldsymbol{A}_{(3,3)} = \begin{bmatrix} 0 & \frac{\omega_z}{2} & -\frac{\omega_y}{2} \\ -\frac{\omega_z}{2} & 0 & \frac{\omega_x}{2} \\ \frac{\omega_y}{2} & -\frac{\omega_x}{2} & 0 \end{bmatrix}$$

$$\boldsymbol{B}_{(3,3)} = \begin{bmatrix} 0 & -g\frac{s_\theta}{\theta} & 0 \\ g\frac{c_\theta s_\phi}{\phi} & 0 & 0 \\ g\frac{(c_\theta+1)(c_\phi-1)}{2\phi} & g\frac{(c_\phi+1)(c_\theta-1)}{2\theta} & 0 \end{bmatrix}$$

$$\boldsymbol{C}_{(3,3)} = \begin{bmatrix} 0 & -\frac{v_z}{2} & \frac{v_y}{2} \\ \frac{v_z}{2} & 0 & \frac{v_x}{2} \\ -\frac{v_y}{2} & \frac{v_x}{2} & 0 \end{bmatrix}$$

$$\boldsymbol{D}_{(3,3)} = \begin{bmatrix} 0 & \frac{(J_y-J_z)\psi}{2J_x} & \frac{(J_y-J_z)\theta}{2J_x} \\ \frac{(J_z-J_x)\psi}{2J_y} & 0 & \frac{(J_z-J_x)\phi}{2J_y} \\ \frac{(J_x-J_y)\theta}{2J_z} & \frac{(J_x-J_y)\phi}{2J_z} & 0 \end{bmatrix}.$$

APPENDIX B
PARAMETER SETTINGS OF THE QUADCOPTER MODEL

Table I shows the values of parameters used in the quadcopter dynamic model in Section IV-A.

REFERENCES

[1] A. Mora, D. F. Glas, T. Kanda, and N. Hagita, "A teleoperation approach for mobile social robots incorporating automatic gaze control and three-dimensional spatial visualization," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 3, pp. 630–642, Feb. 2013.

[2] S. Huang, R. S. H. Teo, and K. K. Tan, "Collision avoidance of multi unmanned aerial vehicles: A review," *Annu. Rev. Control*, vol. 48, pp. 147–164, Jul. 2019.

[3] W. He, T. Meng, X. He, and C. Sun, "Iterative learning control for a flapping wing micro aerial vehicle under distributed disturbances," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1524–1535, Apr. 2018.

[4] L. Kong, W. He, C. Yang, Z. Li, and C. Sun, "Adaptive fuzzy control for coordinated multiple robots with constraint using impedance learning," *IEEE Trans. Cybern.*, vol. 49, no. 3, pp. 3052–3063, Mar. 2019.

[5] W. He, C. Xue, X. Yu, Z. Li, and C. Yang, "Admittance-based controller design for physical human-robot interaction in the constrained task space," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 4, pp. 1937–1949, Oct. 2020.

[6] J. Ma, S.-L. Chen, C. S. Teo, A. Tay, A. A. Mamun, and K. K. Tan, "Parameter space optimization towards integrated mechatronic design for uncertain systems with generalized feedback constraints," *Automatica*, vol. 105, pp. 149–158, May 2019.

[7] J. Ma, Z. Cheng, X. Zhang, M. Tomizuka, and T. H. Lee, "Optimal decentralized control for uncertain systems by symmetric Gauss-Seidel semi-proximal ALM," 2020. [Online]. Available: arXiv:2001.00306.

[8] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," in *Proc. IFAC World Congr.*, 2011, pp. 1485–1491.

[9] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C.-Y. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 6, pp. 740–749, Aug. 2015.

[10] Y. Zhou, H. Hu, Y. Liu, S.-W. Lin, and Z. Ding, "A real-time and fully distributed approach to motion planning for multirobot systems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 23, no. 12, pp. 2636–2650, Dec. 2017.

[11] Z. Chao, L. Ming, Z. Shaolei, and Z. Wenguang, "Collision-free UAV formation flight control based on nonlinear MPC," in *Proc. Int. Conf. Electron. Commun. Control*, 2011, pp. 1951–1956.

[12] N. Deo, A. Rangesh, and M. M. Trivedi, "How would surround vehicles move? A unified framework for maneuver classification and motion prediction," *IEEE Trans. Intell. Veh.*, vol. 3, no. 2, pp. 129–140, Jan. 2018.

[13] N. Ye, Y. Zhang, R. Wang, and R. Malekian, "Vehicle trajectory prediction based on Hidden Markov Model," *KSII Trans. Internet Inf. Syst.*, vol. 10, no. 7, pp. 3150–3170, 2016.

[14] S. Gong, J. Cartlidge, R. Bai, Y. Yue, Q. Li, and G. Qiu, "Extracting activity patterns from taxi trajectory data: A two-layer framework using spatio-temporal clustering, Bayesian probability and Monte Carlo simulation," *Int. J. Geograph. Inf. Sci.*, vol. 34, no. 6, pp. 1–25, 2019.
[15] M. A. Mahmud, M. S. Aman, H. Jiang, A. Abdelgawad, and K. Yelamarthi, "Kalman filter based indoor mobile robot navigation," in *Proc. Int. Conf. Elect. Electron. Optim. Techn.*, 2016, pp. 1949–1953.
[16] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 961–971.
[17] M. Huynh and G. Alaghband, "Trajectory prediction by coupling scene-LSTM with human movement LSTM," in *Proc. Int. Symp. Vis. Comput.*, 2019, pp. 244–259.
[18] M. Assens, X. G.-I. Nieto, K. McGuinness, and N. E. O'Connor, "PathGAN: Visual scanpath prediction with generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 406–422.
[19] D. Roy, T. Ishizaka, C. K. Mohan, and A. Fukuda, "Vehicle trajectory prediction at intersections using interaction based generative adversarial networks," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 2318–2323.
[20] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "SOPHIE: An attentive GAN for predicting paths compliant to social and physical constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1349–1358.
[21] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *Proc. IEEE Intell. Veh. Symp.*, 2012, pp. 141–146.
[22] 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.
[23] R. Pepy and A. Lambert, "Safe path planning in an uncertain-configuration space using RRT," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 5376–5381.
[24] N. Nasios and A. G. Bors, "Variational learning for Gaussian mixture models," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 36, no. 4, pp. 849–862, Jul. 2006.
[25] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer-Verlag, 2006.
[26] W. Härdle and L. Simar, *Applied Multivariate Statistical Analysis*. Berlin, Germany: Springer-Verlag, 2009.
[27] J. Ma *et al.*, "Integrated mechatronic design in the flexure-linked dual-drive gantry by constrained linear–quadratic optimization," *IEEE Trans. Ind. Electron.*, vol. 65, no. 3, pp. 2408–2418, Aug. 2017.
[28] J. Ma *et al.*, "A novel constrained H_2 optimization algorithm for mechatronics design in flexure-linked biaxial gantry," *ISA Trans.*, vol. 71, pp. 467–479, Oct. 2017.
[29] S. Sastry, *Nonlinear Systems: Analysis, Stability, and Control*. New York, NY, USA: Springer-Verlag, 1999.
[30] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.
[31] C. Kirches, *Fast Numerical Methods for Mixed-Integer Nonlinear Model Predictive Control*. Wiesbaden, Germany: Springer, 2011.
[32] E. C. Kerrigan and J. M. Maciejowski, "Soft constraints and exact penalty functions in model predictive control," in *Proc. UKACC Int. Conf. (Control)*, 2000.
[33] X. Zhang, J. Ma, S. Huang, Z. Cheng, and T. H. Lee, "Integrated planning and control for collision-free trajectory generation in 3D environment with obstacles," in *Proc. Int. Conf. Control Autom. Syst.*, 2019, pp. 974–979.
[34] T. Çimen, "State-dependent Riccati equation control: A survey," in *Proc. IFAC World Congr.*, vol. 17, 2018, pp. 3761–3775.
[35] X. Zhang, S. Huang, W. Liang, Z. Cheng, K. K. Tan, and T. H. Lee, "HLT*: Real-time and any-angle path planning in 3D environment," in *Proc. Annu. Conf. IEEE Ind. Electron. Soc.*, 2019, pp. 5231–5236.
[36] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, "MPC-based approach to active steering for autonomous vehicle systems," *Int. J. Veh. Auton. Syst.*, vol. 3, no. 2, pp. 265–291, 2005.

Xiaoxue Zhang received the B.Eng. and M.Sc. degrees in automotive engineering from China Agricultural University, Beijing, China, in 2015 and 2018, respectively. She is currently pursuing the Ph.D. degree in electrical and computer engineering with the National University of Singapore, Singapore.

Her research interests include motion planning, optimal control, robust control, and intelligent transportation systems.

Ms. Zhang is a recipient of the NUS Graduate School for Integrative Sciences and Engineering Scholarship.

Jun Ma received the B.Eng. degree (First Class Hons.) in electrical and electronic engineering from Nanyang Technological University, Singapore, in 2014, and the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2018.

From 2018 to 2019, he was a Research Fellow with the Department of Electrical and Computer Engineering, National University of Singapore. In 2019, he was a Research Associate with the Department of Electronic and Electrical Engineering, University College London, London, U.K. He is currently a Visiting Scholar with the Department of Mechanical Engineering, University of California at Berkeley, Berkeley, CA, USA. His research interests include control theory and optimization, mechatronics and robotics, intelligent transportation systems, and intelligent healthcare.

Dr. Ma was a recipient of the Singapore Commonwealth Fellowship in Innovation.

Zilong Cheng received the B.Eng. degree in automotive engineering from Jilin University, Changchun, China, in 2018. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the National University of Singapore, Singapore.

He has been with the Mechatronics Group, Singapore Institute of Manufacturing Technology, Singapore, since 2018, where he is actively involved in control system design and development. His research interests include control theory and optimization, mechatronics and robotics, and intelligent transportation systems.

Mr. Cheng is a recipient of the NUS Graduate School for Integrative Sciences and Engineering Scholarship.

Sunan Huang received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 1994.

He is currently a Senior Research Scientist with the Temasek Laboratories, National University of Singapore, Singapore. His research interests include coverage and collision avoidance of multiple UAVs as well as fault-tolerant control.

Shuzhi Sam Ge (Fellow, IEEE) received the B.Sc. degree from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 1986, and the Ph.D. degree from Imperial College London, London, U.K., in 1993.

He is the Director of the Social Robotics Laboratory, Interactive Digital Media Institute, Singapore, and the Centre for Robotics, Chengdu, China, and a Professor with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, on leave from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu. He has coauthored four books and over 300 international journal and conference papers. His current research interests include social robotics, adaptive control, intelligent systems, and artificial intelligence.

Dr. Ge is the Editor-in-Chief of the *International Journal of Social Robotics* (Springer). He has served/been serving as an Associate Editor for a number of flagship journals, including IEEE TRANSACTIONS ON AUTOMATION CONTROL, IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, IEEE TRANSACTIONS ON NEURAL NETWORKS, and *Automatica*. He serves as a Book Editor for the *Automation and Control Engineering Series* (Taylor & Francis). He served as the Vice President for Technical Activities from 2009 to 2010 and the Membership Activities from 2011 to 2012 and a member of the Board of Governors from 2007 to 2009 at the IEEE Control Systems Society. He is a Fellow of the International Federation of Automatic Control, the Institution of Engineering and Technology, and the Society of Automotive Engineering.

Tong Heng Lee received the B.A. degree (First Class Hons.) in electrical engineering from the University of Cambridge, Cambridge, U.K., in 1980, the M.Eng. degree in electrical engineering from the National University of Singapore (NUS), Singapore, in 1985, and the Ph.D. degree in electrical engineering from Yale University, New Haven, CT, USA, in 1987.

He is a Professor with the Department of Electrical and Computer Engineering, NUS, and the NUS Graduate School. He was a Past Vice-President (Research) of NUS. His research interests are in the areas of adaptive systems, knowledge-based control, intelligent mechatronics, and computational intelligence.

Prof. Lee is currently an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS, *Control Engineering Practice* (IFAC), and the *International Journal of Systems Science* (Taylor & Francis). He is the Deputy Editor-in-Chief of *Mechatronics* (IFAC).