Gaussian Process Model Predictive Control of Unmanned Quadrotors

Gang Cao, Edmund M-K Lai, Fakhrul Alam
School of Engineering and Advanced Technology, Massey University, Albany
Auckland, New Zealand
Email: {g.cao,e.lai,f.alam}@massey.ac.nz

Abstract Two issues of quadrotor control without deterministic dynamical equations are addressed in this paper by using Gaussian Process (GP) based Model Predictive Control (MPC) algorithm. Firstly, the first issue of modelling unknown dynamical motions is solved by using GP models based on sampled data. In this way, the model uncertainty can be numerically evaluated during modelling and prediction process. This is not easy when using other data-driven methods, such as Artificial Neural Networks (ANN) and Fuzzy Models (FMs). Then a MPC scheme based on obtained GP models is proposed to address the second issue of designing appropriate quadrotor controllers. The proposed algorithm directly takes model uncertainty into account when planning MPC policies, and can be computationally efficiently implemented through using analytical gradients in the optimization process. The performance of quadrotor control using proposed approach is demonstrated by simulations on a trajectory tracking problem.

I. Introduction

Recent interests in quadrotors, which are unmanned aerial vehicles with vertical take-off and landing abilities, are high. This is mainly due to their maneuverability, simplicity and payload capabilities [1]–[3]. They have been proposed for use in various military and civilian tasks [4]–[6].

The dynamics of the quadrotor are highly nonlinear and the control problem of the quadrotor is not trivial. Several control methods have previously been proposed, including sliding model control [7], backstepping control [8] and Model Predictive Control (MPC) [1, 9]. A review of quadrotor control can be found in [9, 10]. MPC has the advantage of conceptual simplicity, and input and output constraints can easily be incorporated.

MPC [11, 12] is a class of computer control algorithms that predict future responses of a plant based on its system model. Control actions are obtained by repeatedly solving a finite horizon optimal control problem. The system model of the quadrotor can be obtained using Newton-Euler [13] or Euler-Lagrange based formalisms [14, 15]. The translational and rotational motions are usually modelled by separately and controlled by two separate controllers [3, 10, 16, 17]. The main drawback of this approach is the difficulty in accounting for unmodelled dynamics and unknown perturbations.

An alternative approach is to use data-driven modelling techniques. Observations or data collected from a quadrotor operating in a real environment can be used to create a model of the dynamics and perturbations through machine learning. In [18], the quadrotor dynamics are modelled by Artificial Neural Networks (ANN) while Fuzzy Models (FMs) are used in [19, 20]. One major issue with these approaches is the difficulty in assessing the quality of the models learnt from data. In [21], a Bayesian based technique is incorporated into ANN to address this issue but it is computationally demanding. Another approach is to use direct Bayesian modelling techniques, such as Gaussian Process (GP) models. GP models have the advantage that variances are computed during the modelling process. These numerical variance values can be used to provide an indication of the quality of models created. Recently, it has been applied to learn partial quadrotor dynamics [22]. Another learning based MPC method can be found in [23].

The main issue with data-driven learning based modelling is that it is generally impossible to account for the full dynamics from the training data alone. Therefore, the main challenge is how to account for model uncertainties. Conventionally, uncertainties are assumed to be bounded, and control actions are computed by using the “min-max” method [24]. However, MPC controllers obtained in this way are usually too conservative since the design is based on worst-case perturbations. Furthermore, uncertainty bounds are not easy to determine in practice. In [25], a Stochastic Model Predictive Control (SMPC) scheme is presented where model uncertainties are represented by probabilistic “hard-constraints”. Unfortunately, this method is computationally demanding. In [26], the use of “soft-constraints” to incorporate model uncertainty into policy planning and evaluation in a straightforward manner has been proposed to address the issue.

In this paper, we consider MPC control of a quadrotor with the full translational and rotational dynamics modelled by using GP. Initially, the quadrotor dynamics are assumed to be totally unknown and a GP model of the dynamics is learnt purely from observations. The advantage of using GP models is that model uncertainties are explicitly expressed by numerically computed variances, and can be propagated over the prediction horizon. We propose a method, called GPMPC to directly take GP model variances into account when designing the controller. In addition, by making use of gradients derived analytically from the GP models, a computationally efficient way is presented to solve the optimization problem.

The rest of this paper is organized as follows. Section II

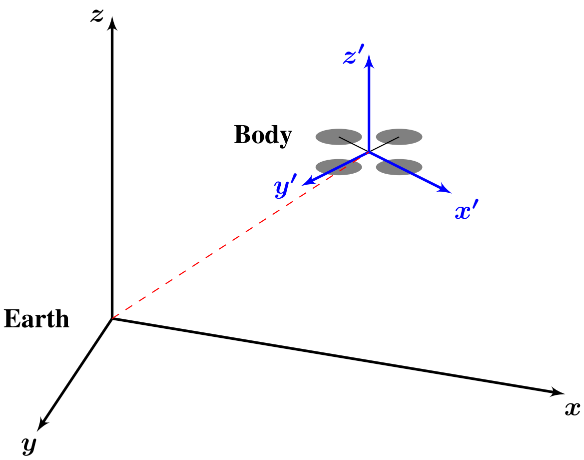


Fig. 1: Quadrotor Body-Earth Frame

briefly introduces Euler-Lagrange based dynamical equations of a quadrotor. The way to learn the GP models of the quadrotor subsystems is presented in Section III. In Section IV, a hierarchical control scheme using GP MPC algorithm is proposed for the trajectory tracking problem. Analytical gradients are derived and they are used in a computationally efficient algorithm to solve the control optimization problem. The effectiveness and efficiency of the proposed algorithm are evaluated through simulation and the results are presented in Section V. Finally, Section VI concludes the paper.

II. Dynamical Models of Quadrotors

The dynamical equations of a quadrotor can be obtained by using the Euler-Lagrange formalism from the energy perspective. As shown in Figure (1), two reference frames are defined – the earth-fixed frame (E-frame) and the body-fixed frame(B-frame). Let $\boldsymbol{\xi}^E[m] = [x, y, z]^T$ and $\boldsymbol{\eta}^E[rad] = [\phi, \theta, \psi]^T$ be the position and angular vectors of the quadrotor in the E-frame and B-frame respectively, where ϕ, θ and ψ are corresponding Euler angles. In addition, let $\mathbf{I} = \text{diag}([I_{xx}, I_{yy}, I_{zz}])$ be the inertia matrix [1], where I_{xx}, I_{yy} and I_{zz} represent the inertial moments w.r.t the corresponding axis.

The Lagrangian of quadrotor dynamics is given by

$$\boldsymbol{\mathcal{L}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = E_T + E_R - E_P \tag{1}$$

where $\boldsymbol{q} = [\boldsymbol{\xi}^E; \boldsymbol{\eta}^E]$ is the generalized coordinate. Here, E_T and E_R denote the translational and rotational kinetic energies, and E_P is the total potential energy. Then, the Euler-Lagrange equation is [10, 14]:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{\Gamma} \end{bmatrix} = \frac{d}{dt}\left(\frac{\partial \boldsymbol{\mathcal{L}}}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial \boldsymbol{\mathcal{L}}}{\partial \boldsymbol{q}} \tag{2}$$

where $\boldsymbol{F}$ is the translational force to the quadrotor, and $\boldsymbol{\Gamma} = [\boldsymbol{\tau}_\phi, \boldsymbol{\tau}_\theta, \boldsymbol{\tau}_\psi]^T$ represent the moments in the roll, pitch and yaw directions.

[1] It is assumed that the quadrotor in this paper is symmetrical w.r.t. all three coordinates (or principal) axes.

Since the Lagrangian does not contain the combined kinetic energy term $\dot{\boldsymbol{q}}$ [10], (2) can be further separated into translational and rotational motion equations, given by

$$\boldsymbol{F} = m\ddot{\boldsymbol{\xi}} + mg\boldsymbol{e_z} \tag{3}$$

where m is mass of the quadrotor and g is gravitational acceleration, $\boldsymbol{e_z} = [0, 0, 1]^T$ is a unit vector in the E-frame, and

$$\boldsymbol{\Gamma} = \boldsymbol{\mathcal{J}}\ddot{\boldsymbol{\eta}}^E + \boldsymbol{\mathcal{C}}\dot{\boldsymbol{\eta}}^E \tag{4}$$

where $\boldsymbol{\mathcal{J}}$ and $\boldsymbol{\mathcal{C}}$ are the Jacobian and Coriolis matrices [10].

By further rewriting (3) in state-space form, the translational subsystem can be expressed as

$$\begin{aligned} \dot{\boldsymbol{x}}_\xi &= f\left(\boldsymbol{x}_\xi, \boldsymbol{u}_\xi\right) + \boldsymbol{\epsilon}_\xi \\ &= \begin{pmatrix} \dot{x} \\ u_x \frac{U_1}{m} + \frac{A_x}{m} \\ \dot{y} \\ u_y \frac{U_1}{m} + \frac{A_y}{m} \\ \dot{z} \\ -g + (\cos\phi\cos\theta)\frac{U_1}{m} + \frac{A_z}{m} \end{pmatrix} + \boldsymbol{\epsilon}_\xi, \end{aligned} \tag{5}$$

with intermediate controls

$$\begin{aligned} u_x &= \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ u_y &= \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \end{aligned} \tag{6}$$

where $\boldsymbol{x}_\xi = [x, \dot{x}, y, \dot{y}, z, \dot{z}]^T$ and $\boldsymbol{u}_\xi = [U_1, u_x, u_y]^T$ are the state and input vectors, and $\boldsymbol{\epsilon}_\xi$ denotes an external disturbance vector. In addition, A_x, A_y and A_z are aerodynamic forces that are independently applied to x, y and z axis in the E-frame.

Similarly, we can obtain the state-space representation of the rotational subsystem as follows.

$$\begin{aligned} \dot{\boldsymbol{x}}_\eta &= g\left(\boldsymbol{x}_\eta, \boldsymbol{u}_\eta\right) + \boldsymbol{\epsilon}_\eta \\ &= \begin{pmatrix} \dot{\phi} \\ \dot{\theta}\dot{\psi}\left(\frac{I_{yy}-I_{zz}}{I_{xx}}\right) + \frac{J_R}{I_{xx}}\dot{\theta}\Omega_R + \frac{L}{I_{xx}}U_2 \\ \dot{\theta} \\ \dot{\phi}\dot{\psi}\left(\frac{I_{zz}-I_{xx}}{I_{yy}}\right) - \frac{J_R}{I_{yy}}\dot{\phi}\Omega_R + \frac{L}{I_{yy}}U_3 \\ \dot{\psi} \\ \dot{\theta}\dot{\phi}\left(\frac{I_{xx}-I_{yy}}{I_{zz}}\right) + \frac{L}{I_{zz}}U_4 \end{pmatrix} + \boldsymbol{\epsilon}_\eta, \end{aligned} \tag{7}$$

where $\boldsymbol{x}_\eta = [\phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi}]^T$ and $\boldsymbol{u}_\eta = [U_2, U_3, U_4]^T$ denote the state and control vectors, and $\boldsymbol{\epsilon}_\eta$ represent the rotational external disturbance vector. In addition, L is arm length of the quadrotor. J_R denotes the inertial moment of rotors, and Ω_R represents the overall residual angular speed of propellers.

III. Quadrotor Dynamics Learning

For both the translational and rotational systems, the state equation take the form:

$$\mathbf{x}_{k+1} = f\left(\mathbf{x}_k, \mathbf{u}_k, k\right) + \mathbf{w}_k \tag{8}$$

where k is the integer index of time, $f(\cdot)$ is an unknown nonlinear function, and $\mathbf{w} \in \mathbb{R}^n$ represents Gaussian noise with zero mean and variance $\boldsymbol{\Sigma}_w$. For the translational system, the states and controls are $\mathbf{x} = \boldsymbol{x}_\xi$ and $\mathbf{u} = \boldsymbol{u}_\xi$ respectively. Similarly, $\mathbf{x} = \boldsymbol{x}_\eta$ and $\mathbf{u} = \boldsymbol{u}_\eta$ for the rotational motions. The system described by (8) can be modelled by GP models where the state-control tuples $\tilde{\mathbf{x}}_k = (\mathbf{x}_k, \mathbf{u}_k) \in \mathbb{R}^{n+m}$ and state differences $\delta\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k \in \mathbb{R}^n$ are used as training inputs and targets respectively [27, 28]. Using state differences can be advantageous when changes in $\delta\mathbf{x}$ are less than changes in $\mathbf{x}$. The n separate GP models are trained for each independent target.

A GP model is completely specified by a mean and a covariance function [29]. If the mean $\boldsymbol{\mu}$ is zero and the squared exponential covariance, defined as $\mathbf{K}(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_j) = \sigma_s^2 \exp(-\frac{1}{2}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)^T \boldsymbol{\Lambda}(\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j)) + \sigma_n^2$, is used, then σ_s^2, σ_n^2 and matrix $\boldsymbol{\Lambda}$ are the hyperparameters of the GP model. Given D training inputs $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \cdots, \tilde{\mathbf{x}}_D]$ and the corresponding training targets $\mathbf{y} = [\delta\mathbf{x}_1, \cdots, \delta\mathbf{x}_D]^T$, the joint distribution between training targets and test target $\delta\mathbf{x}^*$ at a given training input $\tilde{\mathbf{x}}^*$ follows the Gaussian distribution. That is,

$$p\begin{pmatrix} \mathbf{y} \\ \delta\mathbf{x}^* \end{pmatrix} \sim \mathcal{N}\left(0, \begin{array}{cc} \mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}) + \sigma_n \mathbf{I} & \mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{x}}^*) \\ \mathbf{K}(\tilde{\mathbf{x}}^*, \tilde{\mathbf{X}}) & \mathbf{K}(\tilde{\mathbf{x}}^*, \tilde{\mathbf{x}}^*) \end{array}\right) \tag{9}$$

Furthermore, through restricting the joint distribution to only contain those targets that agree with collected observations, we can obtain the posterior distribution that also is a Gaussian with following mean and variance function.

$$\begin{aligned} \mathbb{E}_f[\delta\mathbf{x}_k] &= \mathbf{K}(\tilde{\mathbf{x}}^*, \tilde{\mathbf{X}})(\mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}) + \sigma_n \mathbf{I})^{-1}\mathbf{y} \\ \mathbb{VAR}_f[\delta\mathbf{x}_k] &= \mathbf{K}(\tilde{\mathbf{x}}^*, \tilde{\mathbf{x}}^*) \\ &\quad - \mathbf{K}(\tilde{\mathbf{x}}^*, \tilde{\mathbf{X}})(\mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}) + \sigma_n \mathbf{I})^{-1}\mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{x}}^*) \end{aligned} \tag{10}$$

This equation is typically only used for prediction of the next time-step. When conducting multiple-step predictions, it is necessary to iteratively propagate uncertainties over the prediction horizon. This issue is addressed as predictions with uncertain inputs in [27]. Assuming that the joint distribution of the training input at time k is uncertain and follows a Gaussian distribution $p(\tilde{\mathbf{x}}_k) \sim \mathcal{N}(\tilde{\boldsymbol{\mu}}_k, \tilde{\boldsymbol{\Sigma}}_k)$, the exact predictive distribution of training target can be defined as $p(\delta\mathbf{x}_k) = \int p(f(\tilde{\mathbf{x}}_k)|\tilde{\mathbf{x}}_k)p(\tilde{\mathbf{x}}_k)d\tilde{\mathbf{x}}_k$. This equation is analytically intractable, and normally approximated as a Gaussian with mean $\boldsymbol{\mu}_k^\delta$ and variance $\boldsymbol{\Sigma}_k^\delta$ by using the moment matching technique [27, 30]. This results in

$$\begin{aligned} \boldsymbol{\mu}_k^\delta &= \mathbb{E}_{\tilde{\mathbf{x}}_k}[\mathbb{E}_f[\delta\mathbf{x}_k]] \\ \boldsymbol{\Sigma}_k^\delta &= \begin{bmatrix} \mathbb{VAR}_{f,\tilde{\mathbf{x}}_k}[\delta\mathbf{x}_{k_1}] & \cdots & \mathbb{COV}_{f,\tilde{\mathbf{x}}_k}[\delta\mathbf{x}_{k_n}, \delta\mathbf{x}_{k_1}] \\ \vdots & \ddots & \vdots \\ \mathbb{COV}_{f,\tilde{\mathbf{x}}_k}[\delta\mathbf{x}_{k_1}, \delta\mathbf{x}_{k_n}] & \cdots & \mathbb{VAR}_{f,\tilde{\mathbf{x}}_k}[\delta\mathbf{x}_{k_n}] \end{bmatrix} \end{aligned} \tag{11}$$

The distribution at time $k+1$ can be further approximated by a Gaussian with mean and variance given by

$$\begin{aligned} \boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_k + \boldsymbol{\mu}_k^\delta \\ \boldsymbol{\Sigma}_{k+1} &= \boldsymbol{\Sigma}_k + \boldsymbol{\Sigma}_k^\delta \\ &\quad + \mathbb{COV}_{f,\tilde{\mathbf{x}}_k}[\mathbf{x}_k, \delta\mathbf{x}_k] + \mathbb{COV}_{f,\tilde{\mathbf{x}}_k}[\delta\mathbf{x}_k, \mathbf{x}_k] \end{aligned} \tag{12}$$

More details about the computation of means and variances for uncertain inputs can be found in [27, 31].

Typically, hyperparameters $\boldsymbol{\theta} = [\sigma_s, \sigma_n, \mathrm{vec}(\boldsymbol{\Lambda})]$ are learned by using the evidence maximization technique [32], where $\mathrm{vec}(\cdot)$ denotes vectorization of given matrix. The resulting stochastic optimization problem is conventionally solved by using Conjugate Gradient (CG) or BFGS approaches. More recently, Particle Swarm Optimization (PSO) based algorithms [33] have been proposed to solve this problem.

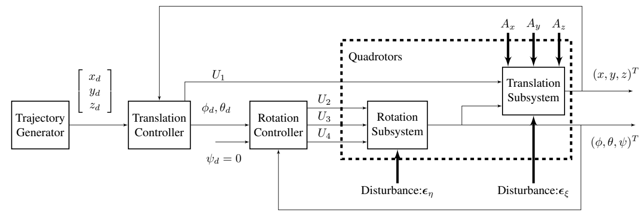


Fig. 2: The Overall Control Scheme for Quadrotors

IV. Trajectory Tracking using GPMPC

A. Overall Control Scheme

The trajectory tracking problem can be tackled by using a hierarchical control structure to orderly handle the tracking problem of the translational subsystem and the corresponding attitude control of the rotational subsystem [3, 10]. The block diagram of this structure is shown in Figure 2. In the outer loop, the translational subsystem is controlled to follow the sequence of desired positions generated by the "Trajectory Generator". The optimal controls U_1 are obtained by minimizing the tracking errors in the "Translation Controller" that also produces desired attitudes θ_d and ϕ_d from intermediate controls u_x and u_y given by (6). Then, the attitudes of the rotational subsystem are tuned to achieve the target values in the inner loop where the desired ψ_d is always set to zero. By minimizing the attitude errors again, the optimal controls U_2, U_3 and U_4 can be obtained from the "Rotation Controller". Finally, those optimal control actions are applied to the quadrotor.

B. Gaussian Process Model Predictive Control

The key issue here is the design of the "Translation Controller" and the "Rotation Controller" given that the translational and rotational subsystems are represented by GP models. We propose a GP based MPC algorithm to address this issue.

Consider an unconstrained MPC control problem of the system given by (8) with the following objective function

$$\mathbf{V}_k^* = \min_{\mathbf{u}(\cdot)} \mathcal{J}(\mathbf{x}_k, \mathbf{u}_{k-1}) \tag{13}$$

where the cost function is given by

$$\begin{aligned} \mathcal{J}(\mathbf{x}_k, \mathbf{u}_{k-1}) = \sum_{i=1}^{H} \big\{ &(\mathbf{x}_{k+i} - \mathbf{r}_{k+i})^T Q (\mathbf{x}_{k+i} - \mathbf{r}_{k+i}) \\ &+ \mathbf{u}_{k+i-1}^T R \mathbf{u}_{k+i-1} \big\} \end{aligned} \tag{14}$$

Here, $\mathbf{r}$ denotes the target positions in translational subsystem, or the target attitudes in rotational subsystem. $Q \in \mathbb{R}^{n \times n}$ and $R \in \mathbb{R}^{m \times m}$ are positive definite weighting matrices, and the prediction horizon H is assumed to be same as the control horizon. In addition, because $\mathbf{x}_k$ are GP predictions, (13) actually becomes a stochastic one [25, 34].

$$\mathbf{V}_k^* = \min_{\mathbf{u}(\cdot)} \mathbb{E}\left[\mathcal{J}(\mathbf{x}_k, \mathbf{u}_{k-1})\right] \tag{15}$$

The expected value of the cost function can be derived as

$$\begin{aligned}\mathbb{E}\left[\mathcal{J}(\mathbf{x}_k,\mathbf{u}_{k-1})\right] &= \mathbb{E}\left[\sum_{i=1}^{H}\left\{(\mathbf{x}_{k+i}-\mathbf{r}_{k+i})^T Q(\mathbf{x}_{k+i}-\mathbf{r}_{k+i})\right.\right.\\ &\quad \left.\left.+\mathbf{u}_{k+i-1}^T R\mathbf{u}_{k+i-1}\right\}\right]\\ &= \sum_{i=1}^{H}\mathbb{E}\left[(\mathbf{x}_{k+i}-\mathbf{r}_{k+i})^T Q(\mathbf{x}_{k+i}-\mathbf{r}_{k+i})\right.\\ &\quad \left.+\mathbf{u}_{k+i-1}^T R\mathbf{u}_{k+i-1}\right]\end{aligned} \tag{16}$$

Since the values of the controls are deterministic in practice, the joint distribution of the state-control tuple at sample time k can be expressed by

$$\begin{aligned}p(\tilde{\mathbf{x}}_k) &= p\begin{pmatrix}\mathbf{x}_k\\ \mathbf{u}_k\end{pmatrix}\\ &\sim\mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_k\\ \mathbf{u}_k\end{bmatrix},\begin{bmatrix}\boldsymbol{\Sigma}_k & \mathbb{COV}[\mathbf{x}_k,\mathbf{u}_k]\\ \mathbb{COV}[\mathbf{u}_k,\mathbf{x}_k] & \mathbb{COV}[\mathbf{u}_k,\mathbf{u}_k]\end{bmatrix}\right)\end{aligned} \tag{17}$$

where $\mathbb{COV}[\mathbf{x}_k,\mathbf{u}_k], \mathbb{COV}[\mathbf{u}_k,\mathbf{x}_k]$ and $\mathbb{COV}[\mathbf{u}_k,\mathbf{u}_k]$ are zero. Thus the cost function (16) can be simplified to

$$\begin{aligned}\mathbb{E}\left[\mathcal{J}(\mathbf{x}_k,\mathbf{u}_{k-1})\right] &= \sum_{i=1}^{H}\left\{(\boldsymbol{\mu}_{k+i}-\mathbf{r}_{k+i})^T Q(\boldsymbol{\mu}_{k+i}-\mathbf{r}_{k+i})\right.\\ &\quad \left.+\text{trace}(Q\boldsymbol{\Sigma}_{k+i})+\mathbf{u}_{k+i-1}^T R\mathbf{u}_{k+i-1}\right\}\end{aligned} \tag{18}$$

This simplification essentially transformed the stochastic cost function into a deterministic one. Therefore most linear and nonlinear optimization methods can be used to solve the problem. In addition, the propagated uncertainties are included in the cost function. This provides a straightforward way to compute desired controls with the consideration of model uncertainties.

C. Gradient Based Optimization

Solving (15) is computationally demanding. The computational complexity of the one-step moment matching in (11) alone requires $\mathcal{O}(D^2n^2(n+m))$ operations. With the complexities of hyperparameters learning, i.e. $\mathcal{O}(nD^3)$, only problems with limited dimensions (under 12 as suggested by most publications) and limited size of training data can make use of GP based MPC. In this section, we shall describe our gradient-based algorithm that is significantly less demanding computationally.

Assuming $h(z)=\mathbb{E}[\mathcal{J}(\mathbf{x}_k,\mathbf{u}_{k-1})]$, the optimization problem (15) can be expressed more compactly as

$$z^* = \arg\min_{z\in\mathbb{Z}} h(z) \tag{19}$$

$h(\cdot)$ is a value-based differentiable function over the whole solution domain $\mathbb{Z}\subseteq\mathbb{R}^m$. z^* denotes an optimal solution that satisfies $\nabla_z h(z^*)=0$ and $\nabla_z^2 h(z^*)\geq 0$. Since the optimization approaches using second-order derivatives $\nabla_z^2 h(\cdot)$, such as Newton's method, can improve accuracy but is computational demanding, we only use first-order derivatives $\nabla_z h(\cdot)$. Note that both derivatives are available when using GP models [27].

Input: Learning GP Models, H, $\mathbf{r}_k$, Q, R.
Initialization:
Maximum iterations $N=1000$,
$\epsilon = 1.0\times 10^{-6}$,
initial inputs $\mathbf{u}_0$ and optimal controls $\mathbf{u}^*=\mathbf{u}_0$;
for $i=1$ *to* N **do**
  if $\mathbb{E}[\mathcal{J}(\mathbf{u}_i)]\leq\epsilon$ **then**
    $\mathbf{u}^*=\mathbf{u}_i$;
    End Loop;
  else
    Caculate graidents $\frac{d\mathbb{E}[\mathcal{J}(\mathbf{u}_i)]}{d\mathbf{u}_{i-1}}$ using (23);
    Update step length α_s accroding to [35];
    Update controls $\mathbf{u}_{i+1}=\mathbf{u}_i+\alpha_s\frac{d\mathbb{E}[\mathcal{J}(\mathbf{u}_i)]}{d\mathbf{u}_{i-1}}$;
    i=i+1;
  end
end
Output: Optimal controls $\mathbf{u}^*$.

Algorithm 1: Analytical gradient based optimization method

The optimal solution z^* can be obtained by iteratively conducting a linear or steepest descent search where

$$z(i+1) = z(i) + \alpha_s\nabla_z h(z(i)) \tag{20}$$

with initial guess $z_0\subseteq\mathbb{R}^m$ until one that satisfies $h(z(i)) - h(z^*)\geq\epsilon$ is reached. Here, ϵ is a predefined tolerance, and α_s is the search step size. Using this method, suboptimal solutions can still be found even if the problem is non-convex.

The key issue in implementing this gradient-based method on problem (15) is computing the gradients that are derivatives of the value function w.r.t. controls. Numerical methods such as finite difference are often used to approximate the gradients [36]. They are easy to implement but may lead to poor gradients due to the nature of the approximation methods [37]. With the use of GP models, the gradients can be readily obtained analytically without the need for numerical approximations.

Let

$$\begin{aligned}\mathcal{H}_i =& (\boldsymbol{\mu}_{k+i}-\mathbf{r}_{k+i})^T Q(\boldsymbol{\mu}_{k+i}-\mathbf{r}_{k+i})\\ &+\text{trace}(Q\boldsymbol{\Sigma}_k)+\mathbf{u}_{k+i-1}^T R\mathbf{u}_{k+i-1}\end{aligned} \tag{21}$$

Then from (18), $\mathbb{E}\left[\mathcal{J}(\mathbf{x}_k,\mathbf{u}_{k-1})\right]=\sum_{i=1}^{H}\mathcal{H}_i$. The gradients can be expressed, using the chain-rule, as

$$\frac{d}{d\mathbf{u}_{k-1}}\mathbb{E}\left[\mathcal{J}(\mathbf{x}_k,\mathbf{u}_{k-1})\right]=\sum_{i=1}^{H}\frac{d\mathcal{H}_i}{d\mathbf{u}_{k+i-1}} \tag{22}$$

and

$$\begin{aligned}\frac{d\mathcal{H}_i}{d\mathbf{u}_{k+i-1}} =& \frac{\partial\mathcal{H}_i}{\partial\boldsymbol{\mu}_{k+i}}\frac{\partial\boldsymbol{\mu}_{k+i}}{\partial\mathbf{u}_{k+i-1}}\\ &+\frac{\partial\mathcal{H}_i}{\partial\boldsymbol{\Sigma}_{k+i}}\frac{\partial\boldsymbol{\Sigma}_{k+i}}{\partial\mathbf{u}_{k+i-1}}+\frac{\partial\mathcal{H}_i}{\partial\mathbf{u}_{k+i-1}}\end{aligned} \tag{23}$$

where $\frac{\partial\mathcal{H}_k}{\partial\boldsymbol{\mu}_k}$, $\frac{\partial\mathcal{H}_k}{\partial\boldsymbol{\Sigma}_k}$ and $\frac{\partial\mathcal{H}_k}{\partial\mathbf{u}_{k-1}}$ can be easily obtained. Also,

$$\begin{aligned}\frac{\partial\boldsymbol{\mu}_{k+i}}{\partial\mathbf{u}_{k+i-1}} &= \frac{\partial\boldsymbol{\mu}_{k+i}}{\partial\tilde{\boldsymbol{\mu}}_{k+i-1}}\frac{\partial\tilde{\boldsymbol{\mu}}_{k+i-1}}{\partial\mathbf{u}_{k+i-1}}\\ \frac{\partial\boldsymbol{\Sigma}_{k+i}}{\partial\mathbf{u}_{k+i-1}} &= \frac{\partial\boldsymbol{\Sigma}_{k+i}}{\partial\tilde{\boldsymbol{\Sigma}}_{k+i-1}}\frac{\partial\tilde{\boldsymbol{\Sigma}}_{k+i-1}}{\partial\mathbf{u}_{k+i-1}}\end{aligned} \tag{24}$$

where $\frac{\partial \tilde{\boldsymbol{\mu}}_{k+i-1}}{\partial \mathbf{u}_{k+i-1}}$ and $\frac{\partial \tilde{\boldsymbol{\Sigma}}_{k+i-1}}{\partial \mathbf{u}_{k+i-1}}$ can be easily obtained as well. Algorithm 1 summarizes our gradient based algorithm for the optimization problem in each iteration of the MPC.

V. Numerical Simulations

The performance of the proposed approach to trajectory tracking is evaluated by computer simulation. A "Lorenz" trajectory (shown as red dotted line in Figure 5) is used in the presence of external Gaussian white noise with zero mean and unit variance. The aerodynamic forces and moments, as well as other parameters used in translational and rotational dynamical equations are the same as those used in [3]. All simulations are conducted 50 times on a computer with a 3.40GHz Intel® Core™ 2 Duo CPU with 16 GB RAM, using Matlab® version 8.1.

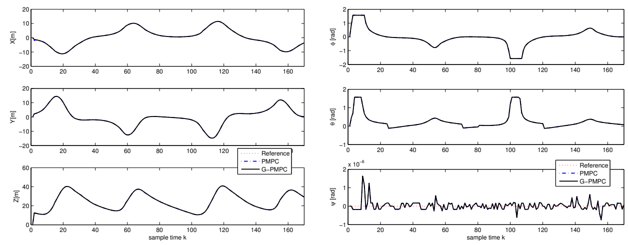
To collect training data, we use the hierarchical scheme shown in Figure 2 but based on deterministic dynamical models (5) and (7), and the "min-max" Nonlinear Model Predictive Control (NMPC) method proposed in [24]. 170 observations including states and controls are used to train the GP models of two subsystems. For the rotational subsystem, the data are scaled to the range $[0.1, 0.9]$. This is necessary mainly due to the large numerical ranges in the original data. For example, the unscaled angle ϕ lies in the range $[-1.57, 1.57]$ while input U_4 lies in the range $[-3.2, 6.2] \times 10^{-8}$. The scaled data leads to much improved training results. The training of all GP models takes 4.5 seconds. The Mean Squared Error (MSE) values of obtained models for two subsystems are very small, i.e. 1.8693×10^{-7} and $8.5238e \times 10^{-9}$ respectively for the translational and rotational models. This implies that the models are well trained.

These GP models are used to predict future quadrotor positions and attitudes in the process of designing "Translation Controller" and "Rotation Controller". Theoretically, a "sufficiently long" prediction horizon H is required to guarantee stability of the MPC scheme [38]. Here, we use the shortest prediction and control horizon of 1 to test our control method at extreme conditions. The positions and attitudes produced by using GPMPC schemes are shown in Figures 3a and 3b. The "Reference" is the Lorenz trajectory. "G-PMPC" denotes the GPMPC solved by our analytical gradient based algorithm, and "PMPC" denotes the one solved without using gradient information. These two figures show that both "PMPC" and "G-PMPC" closely follow the reference positions and attitudes over the whole trajectory. The tracking MSE given in Table I shows that "PMPC" generally produces slightly better results than "G-PMPC". These are to be expected as they solved the same problem. The difference lies mainly in the computational efficiency. "G-PMPC" takes only 27% and 32% of the time required by "PMPC" for position and attitude control respectively while producing very similar control performances. This shows that the gradient based algorithm is both efficient and effective.

The overall trajectory tracking results are depicted in Figure 5.

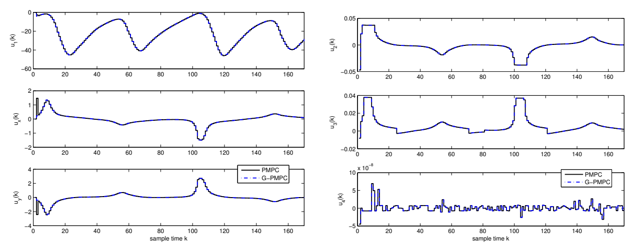
TABLE I: MSE values of position tracking and attitude control using GP based MPC schemes in the "Lorenz" trajectory tracking problem.

	MSE Values	
	G-PMPC	PMPC
Position X	3.3418×10^{-4}	4.3401×10^{-4}
Position Y	5.1399×10^{-5}	1.6264×10^{-5}
Position Z	0.0010	0.0010
Attitude ϕ	1.5743×10^{-8}	4.3030×10^{-9}
Attitude θ	5.5213×10^{-9}	2.5044×10^{-9}
Attitude ψ	6.4365×10^{-14}	6.4368×10^{-14}



(a) Controlled positions x,y,z (b) Controlled attitudes ϕ, θ, ψ

Fig. 3: Controlled positions and attitudes in the "Lorenz" tracking task.



(a) Position controls U_1, u_x, u_y (b) Attitude controls U_2, U_3, U_4

Fig. 4: Obtained position and attitude controls of using GP model based MPC schemes.

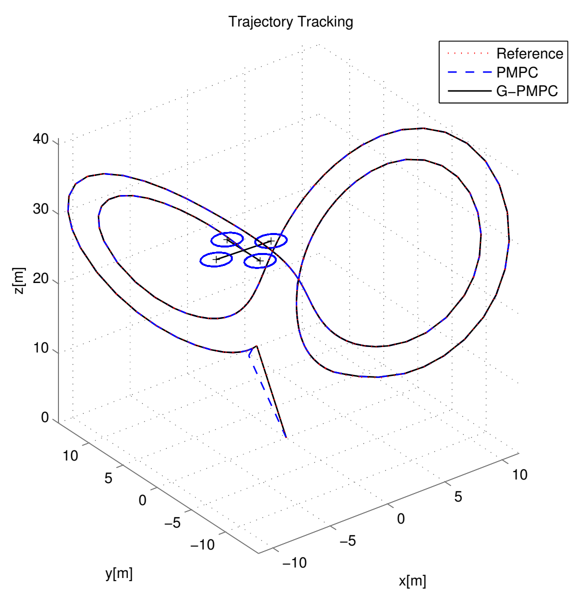


Fig. 5: "Lorenz" trajectory tracking.

VI. CONCLUSIONS

A GP based MPC strategy is proposed for the trajectory tracking problem of a quadrotor. The overall control structure is a hierarchical scheme that consists of two separate MPC controllers for the translational and rotational subsystem respectively. GP models of the dynamics of these two subsystems are learnt from empirical data. The GPMPC scheme is able to account for model uncertainties when computing MPC controls. In addition, a computationally efficient analytical gradient based algorithm to solve the GPMPC optimization problem is proposed. Simulation results show that the GPMPC is able to track a non-trivial trajectory very well. They also show that the analytical gradient based algorithm significantly reduces computational demand in solving the optimization problem.

REFERENCES

[1] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances," *Control Engineering Practice*, vol. 19, no. 10, pp. 1195–1207, 2011.

[2] M. Abdolhosseini, Y. Zhang, and C. A. Rabbath, "An efficient model predictive control scheme for an unmanned quadrotor helicopter," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 27–38, 2013.

[3] K. Alexis, G. Nikolakopoulos, and A. Tzes, "On trajectory tracking model predictive control of an unmanned quadrotor helicopter subject to aerodynamic disturbances," *Asian Journal of Control*, vol. 16, no. 1, pp. 209–224, 2014.

[4] N. Metni and T. Hamel, "A UAV for bridge inspection: Visual servoing control law with orientation limits," *Automation in construction*, vol. 17, no. 1, pp. 3–10, 2007.

[5] P. Doherty and P. Rudol, "A UAV search and rescue scenario with human body detection and geolocalization," in *Advances in Artificial Intelligence*. Springer, 2007, pp. 1–13.

[6] K. Alexis, G. Nikolakopoulos, A. Tzes, and L. Dritsas, "Coordination of helicopter UAVs for aerial forest-fire surveillance," in *Applications of intelligent control to engineering systems*. Springer, 2009, pp. 169–193.

[7] T. Madani and A. Benallegue, "Sliding mode observer and backstepping control for a quadrotor unmanned aerial vehicles," in *American Control Conference*. IEEE, 2007, pp. 5887–5892.

[8] M. Huang, B. Xian, C. Diao, K. Yang, and Y. Feng, "Adaptive tracking control of underactuated quadrotor unmanned aerial vehicles via backstepping," in *American Control Conference*. IEEE, 2010, pp. 2076–2081.

[9] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Model predictive quadrotor control: attitude, altitude and position experimental studies," *IET Control Theory & Applications*, vol. 6, no. 12, pp. 1812–1827, 2012.

[10] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "An integral predictive/nonlinear H control structure for a quadrotor helicopter," *Automatica*, vol. 46, no. 1, pp. 29–39, 2010.

[11] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*. Springer, 1999, pp. 207–226.

[12] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[13] Z. Zuo, "Trajectory tracking control design with command-filtered compensation for a quadrotor," *IET Control Theory & Applications*, vol. 4, no. 11, pp. 2343–2355, 2010.

[14] S. Bouabdallah, A. Noth, and R. Siegwart, "PID vs LQ control techniques applied to an indoor micro quadrotor," in *IEEE/RSJ Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, vol. 3. IEEE, 2004, pp. 2451–2456.

[15] P. Castillo, A. Dzul, and R. Lozano, "Real-time stabilization and tracking of a four-rotor mini rotorcraft," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 4, pp. 510–516, 2004.

[16] A. Das, F. Lewis, and K. Subbarao, "Backstepping approach for controlling a quadrotor using lagrange form dynamics," *Journal of Intelligent and Robotic Systems*, vol. 56, no. 1-2, pp. 127–151, 2009.

[17] L. Wang and H. Jia, "The trajectory tracking problem of quadrotor UAV: Global stability analysis and control design based on the cascade theory," *Asian Journal of Control*, vol. 16, no. 2, pp. 574–588, 2014.

[18] T. Dierks and S. Jagannathan, "Output feedback control of a quadrotor UAV using neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 50–66, 2010.

[19] C. Coza and C. Macnab, "A new robust adaptive-fuzzy control method applied to quadrotor helicopter stabilization," in *Annual meeting of the North American Fuzzy Information Processing Society, NAFIPS 2006*. IEEE, 2006, pp. 454–458.

[20] Z. Lendek, A. Berna, J. Guzmán-Giménez, A. Sala, and P. García, "Application of Takagi-Sugeno observers for state estimation in a quadrotor," in *IEEE Proceedings of International Conference on Decision and Control*. IEEE, 2011, pp. 7530–7535.

[21] F. Aires, C. Prigent, and W. B. Rossow, "Neural network uncertainty assessment using Bayesian statistics: A remote sensing application," *Neural computation*, vol. 16, no. 11, pp. 2415–2458, 2004.

[22] F. Berkenkamp and A. P. Schoellig, "Safe and robust learning control with Gaussian processes," in *European Control Conference*, 2015, pp. 2501–2506.

[23] A. Aswani, P. Bouffard, and C. Tomlin, "Extensions of learning-based model predictive control for real-time application to a quadrotor helicopter," in *American Control Conference*. IEEE, 2012, pp. 4661–4666.

[24] P. Scokaert and D. Mayne, "Min-max feedback model predictive control for constrained linear systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 8, pp. 1136–1142, 1998.

[25] A. Grancharova, J. Kocijan, and T. A. Johansen, "Explicit stochastic predictive control of combustion plants based on Gaussian process models," *Automatica*, vol. 44, no. 6, pp. 1621–1631, 2008.

[26] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and A. Girard, "Gaussian process model based predictive control," in *American Control Conference*, vol. 3. IEEE, 2004, pp. 2214–2219.

[27] M. P. Deisenroth, "Efficient reinforcement learning using Gaussian processes," Ph.D. dissertation, Karlsruhe Institute of Technology, 2010.

[28] Y. Pan and E. Theodorou, "Probabilistic differential dynamic programming," in *Advances in Neural Information Processing Systems*, 2014, pp. 1907–1915.

[29] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 1 2006.

[30] J. Q. Candela, A. Girard, J. Larsen, and C. E. Rasmussen, "Propagation of uncertainty in bayesian kernel models-application to multiple-step ahead forecasting," in *IEEE Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2. IEEE, 2003, pp. II–701.

[31] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408–423, 2015.

[32] D. J. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[33] G. Cao, E. M. Lai, and F. Alam, "Particle swarm optimization for convolved Gaussian process models," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 6-11 July 2014, pp. 1573–1578.

[34] B. Kouvaritakis and M. Cannon, "Stochastic model predictive control," *Encyclopedia of Systems and Control*, pp. 1–9, 2014.

[35] Y. Yuan, "Step-sizes for the gradient method," *AMS IP Studies in Advanced Mathematics*, vol. 42, no. 2, p. 785, 2008.

[36] C. Kirches, *Fast numerical methods for mixed-integer nonlinear model-predictive control*. Springer, 2011.

[37] L. Imsland, P. Kittilsen, and T. S. Schei, "Model-based optimizing control and estimation using Modelica model," *Modeling, Identification and Control*, vol. 31, no. 3, pp. 107–121, 2010.

[38] A. Jadbabai and J. Hauser, "On the stability of receding horizon control with a general terminal cost," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 674–678, 2005.